2004年度

オープンソースソフトウェア活用基盤整備事業

WfMC 標準準拠のワークフローシステム運用基盤の開発

基 本 設 計 書

－ Version 0.97 －

Ver0.9 2005 年 3 月 1 日
Ver0.91 2005 年 3 月 15 日
Ver0.92 2005 年 4 月 30 日
Ver0.93 2005 年 5 月 12 日
Ver0.94 2005 年 5 月 16 日
Ver0.95 2005 年 5 月 24 日
Ver0.96 2005 年 5 月 27 日
Ver0.97 2005 年 6 月 6 日

株式会社アルゴ21

変更履歴

版	更新者	更新日	更新箇所	更　　新　　内　　容
0.9	伊藤	2005/03/01		新規作成
0.91	伊藤	2005/03/15	別紙参照	ワークフロー基本設計書レビュー議事録20050309.doc を反映
0.91	伊藤	2005/04/15	別紙参照	ワークフロー基本設計書レビュー議事録20050328.doc を反映
0.92	内田	2005/04/21	別紙参照	文言修正
0.93	伊與田	2005/05/12	Ⅲ .2. DB テーブル仕様	履歴テーブルの主キー指定を削除
0.94	伊與田	2005/05/16	Ⅲ .2. DB テーブル仕様	プロセス変数変更履歴テーブルにACT_ID カラムを追加
0.95	伊與田	2005/05/24	Ⅲ .2. DB テーブル仕様	頂点組織別名テーブル削除 組織テーブル責任者 ID カラム削除
0.96	伊與田	2005/05/27	Ⅲ .2. DB テーブル仕様	アプリケーションタイプテーブル削除 アプリケーションテーブル削除 セッション変更履歴テーブル削除
0.97	真鍋	2005/06/06	Ⅱ.機能仕様	WorkItemHandler の reassignWorkItem の引数にソース参加者を追加 エンジン間接続インタフェースに disconnect を追加
0.98	伊藤	2005/9/30	Ⅱ.機能仕様	状態遷移管理機能の図を更新

レビュー実施履歴

対象版	実施日	承認者	参　　加　　者	更新版
0.9	2005/03/09 ～03/10	服部	服部、小林、伊與田、内田（玲）、伊藤	0.91
0.9	2005/03/28 ～03/29	服部	服部、小林、伊與田、内田（玲）、伊藤	0.91

目　　次

I　システム概要編 1
1　開発の背景 2
2　開発の目的 3
2.1　現状の問題点 3
2.2　問題解決策 3
3　本システムの活用イメージ 5
4　動作イメージ 8
5　システム構成 9
5.1　システム全体概要図および実現範囲 9
5.2　ハードウェア要件 11
5.3　ソフトウェア要件 11
5.4　ネットワーク要件 12
5.5　配置形態 12
6　検収条件 13
6.1　機能的検収条件 13
6.2　性能的検収条件 15
6.3　普及活動等に向けての行動および具備すべき条件 15
7　ライセンス 16
II　機能仕様編 17
1　機能構成図 18
2　モジュール構成 19
3　基本定義要素 20
3.1　ワークフロー定義 20
3.1.1　XPDL の定義構成 20
3.1.2　クラス図 21
3.2　パッケージ定義 21
3.2.1　概要 22
3.2.2　インタフェース 22
3.2.3　XML スキーマ定義と XML 表現 22
3.3　ワークフロー参加者定義 23
3.3.1　概要 23
3.3.2　インタフェース 23
3.3.3　XML スキーマ定義と XML 表現 23
3.4　ワークフローアプリケーション定義 24
3.4.1　概要 24

3.4.2 インタフェース 24
3.4.3 XML スキーマ定義と XML 表現 24
3.5 プロセス変数定義 25
3.5.1 概要 25
3.5.2 インタフェース 25
3.5.3 XML スキーマ定義と XML 表現 25
3.6 プロセス定義 26
3.6.1 概要 26
3.6.2 インタフェース 26
3.6.3 XML スキーマ定義と XML 表現 27
3.7 アクティビティ定義 28
3.7.1 概要 28
3.7.2 インタフェース 28
3.7.3 XML スキーマ定義と XML 表現 29
3.8 遷移定義 30
3.8.1 概要 30
3.8.2 インタフェース 30
3.8.3 XML スキーマ定義と XML 表現 31
3.9 パラメータ定義 31
3.9.1 概要 31
3.9.2 インタフェース 31
3.9.3 XML スキーマ定義と XML 表現 32
3.10 条件式 32
3.10.1 クラス図 32
3.10.2 XML スキーマ定義と XML 表現 33
3.10.3 条件判定の流れ 34
4 フィルター 36
4.1 クラス図 36
4.2 Filter 36
4.2.1 概要 36
4.2.2 インタフェース 37
5 定義リポジトリ 38
5.1 ワークフロー定義とインスタンスの関係 38
5.1.1 作業項目の収集手順 39
5.2 ワークフロー定義ファイルの配置 39
5.3 クラス図 40

5.4 DefinitionRepository 40
5.4.1 概要 40
5.4.2 インタフェース 41
6 ワークフローエンジンのコアインタフェース 42
6.1 クラス図 42
6.2 WfObject 42
6.2.1 概要 42
6.3 WfExecutionObject 42
6.3.1 概要 42
6.3.2 インタフェース 43
6.4 WfProcess 44
6.4.1 概要 44
6.4.2 インタフェース 44
6.5 WfActivity 45
6.5.1 概要 45
6.5.2 インタフェース 46
6.6 WfEvent 46
6.6.1 概要 46
6.6.2 インタフェース 47
6.7 WfRequester 47
6.7.1 概要 47
6.7.2 インタフェース 48
6.8 WfResource 48
6.8.1 概要 48
6.8.2 インタフェース 48
6.9 WfAssignment 49
6.9.1 概要 49
6.9.2 インタフェース 49
6.10 WfProcessManager 50
6.10.1 概要 50
6.10.2 インタフェース 50
7 ワークフローサービス管理 52
7.1 クラス図 52
7.2 WorkflowServiceManager 52
7.2.1 概要 52
7.2.2 インタフェース 52

7.3 WorkflowEngineHandler 52
7.3.1 概要 52
7.3.2 インタフェース 53
7.4 WorkItemHandler 54
7.4.1 概要 55
7.4.2 インタフェース 55
7.5 WorkflowAdminHandler 56
7.5.1 概要 56
7.5.2 インタフェース 56
7.6 WorkItem 57
7.6.1 概要 57
7.6.2 インタフェース 57
8 ツールエージェント 58
8.1 クラス図 58
8.2 ToolAgentFactory 58
8.2.1 概要 58
8.2.2 インタフェース 58
8.3 ToolAgent 58
8.3.1 概要 58
8.3.2 インタフェース 59
8.3.3 設定ファイル 59
9 エンジン間接続 61
9.1 概念図 61
9.2 クラス図 61
9.3 InterWorkflowConnector 61
9.3.1 概要 61
9.3.2 インタフェース 61
9.3.3 設定ファイル 63
10 ワークフロー参加者管理 64
10.1 クラス図 64
10.2 ResourceManagerFactory 64
10.2.1 概要 64
10.2.2 インタフェース 64
10.3 UserManager 65
10.3.1 概要 65
10.3.2 インタフェース 65

10.4 OrganizationManager 65
10.4.1 概要 65
10.4.2 インタフェース 65
10.5 RoleManager 66
10.5.1 概要 66
10.5.2 インタフェース 66
10.6 User 67
10.6.1 概要 67
10.6.2 インタフェース 67
10.7 Organization 67
10.7.1 概要 68
10.7.2 提供インタフェース 68
10.8 Role 68
10.8.1 概要 68
10.8.2 提供インタフェース 68
11 セッション管理 69
11.1 クラス図 69
11.2 SessionManagerFactory 69
11.2.1 概要 69
11.2.2 提供インタフェース 69
11.3 SessionManager 70
11.3.1 概要 70
11.3.2 提供インタフェース 70
11.4 Session 71
11.4.1 概要 71
11.4.2 インタフェース 71
11.5 SimpleSessionManager 71
11.5.1 概要 71
11.6 SessionPrincipal 71
11.6.1 概要 72
11.6.2 提供インタフェース 72
12 ロギング 73
12.1 概要 73
13 各機能説明 74
13.1 ワークフローエンジン 74
13.1.1 処理要求管理機能 74

13.1.2 状態遷移管理機能 75
13.1.3 アプリケーション実行制御機能 95
13.1.4 XPDL 定義ファイル解析機能 95
13.2 ワークフローAPI 97
13.2.1 運用管理用 API 99
13.2.2 クライアント用 API 99
13.2.3 アプリケーション連携用 API 100
13.2.4 ワークフローエンジン間連携用 API 100
13.3 ワークフロー定義ツール 100
13.4 ワークフロー運用管理ツール 101
13.5 通信インタフェース 101
13.5.1 RMI over IIOP 101
13.5.2 Web サービス 101
13.5.3 Servlet 102
13.6 サンプルアプリケーション 102
III 入出力データ編 103
1 ワークフロー定義仕様 104
1.1 XPDL スキーマ仕様 104
1.2 遷移条件式スキーマ仕様 115
2 DB テーブル仕様 117
2.1 テーブル一覧 117
2.2 テーブル詳細 117
2.2.1 キーシーケンス管理（KEYSEQ） 117
2.2.2 ワークフロー定義管理（DEFINITION） 117
2.2.3 組織(ORGANIZATION) 118
2.2.4 ユーザ(USER) 118
2.2.5 役割(ROLE) 118
2.2.6 組織ユーザ関連(ORGUSERMAP) 118
2.2.7 組織役割関連(ORGROLEMAP) 118
2.2.8 ユーザ役割関連(USERROLEMAP) 118
2.2.9 プロセスインスタンス(PROCESS) 118
2.2.10 プロセス変数インスタンス（VARIABLE） 119
2.2.11 アクティビティインスタンス(ACTIVITY) 119
2.2.12 作業項目(WORKITEM) 120
2.2.13 プロセス完了待ちアクティビティ(WAITPROCACT) 120
2.2.14 並行処理ブロック(CONCURBLOCK) 120

2.2.15 アプリケーション実行(APPEXEC) 121
2.2.16 プロセス状態変更履歴(PROCESSHIST) 121
2.2.17 プロセス変数変更履歴（VARIABLEHIST） 121
2.2.18 アクティビティ状態変更履歴(ACTIVITYHIST) 122
2.2.19 起動先プロセス(ETAPROCESS) 122
2.2.20 起動元アクティビティ(ETAACTIVITY) 123
IV 資料編 124
1 商用ワークフローシステム運用基盤一覧 125
2 主な OSS ワークフローシステム運用基盤ソフトウェアの比較 126
3 WfMC のワークフロー参照モデルと策定状況 127
4 WfMC ワークフローAPI 対応表 129
4.1 インタフェース１ 129
4.2 インタフェース２＆３ 129
4.2.1 基本データ型 129
4.2.2 その他のデータ型 130
4.2.3 属性 133
4.2.4 エラーコード 134
4.2.5 ワークフローAPI 135
4.3 インタフェース４ 146
4.4 インタフェース５ 148
4.4.1 監査データ型定義 148
4.4.2 CWAD 接頭情報 149
4.4.3 CWAD 接尾情報 149
4.4.4 プロセスインスタンス監査情報 149
4.4.5 アクティビティインスタンス監査情報 150
4.4.6 作業項目監査情報 150
4.4.7 リモート操作情報 151

Ⅰ　システム概要編

1　開発の背景

インターネットの普及に伴い、得意分野を生かした企業間の連携や、企業内でも部門間の業務システムの連動により、経営の意志決定速度を向上させたいというニーズが高まっている。その際、既存システムに対する変更をいかに少なくするかが、システム構築に当たっての課題の一つである。その解決策として、データを中心として既存システムを緩やかに統合することが注目されている。そして、ワークフローエンジンを中心としたワークフローシステム運用基盤は、その実現のための重要な構成要素と認識されている。

しかしながら、商用のワークフローシステム運用基盤は、単体で 100 万円～1,000 万円と非常に高額なものが多く、これが中堅・中小企業への普及を妨げる大きな要因となっている（主な商用製品の参考価格については、資料編１を参照のこと）。また、商用のワークフローシステム運用基盤は自社製のAPサーバやEJBコンテナ 等特定の環境に依存するものが多く、適用は比較的大規模なシステムに限定される傾向にある。

オープンソースソフトウェア（以下、OSS）に活路を求めてみると、海外には OSS のワークフローシステム運用基盤がいくつか存在するが、いずれも日本での認知度は非常に低く、活用というレベルまでいたっていない（代表的な OSS ワークフローシステム運用基盤については、資料編２を参照のこと）。

日本で OSS ワークフローシステム運用基盤が注目されない理由として大きく二つの原因が考えられる。一つには日本においては「ワークフロー＝文書承認システム」というイメージが定着しているため、ミドルウェアとして提供されるワークフローシステム運用基盤の活用イメージに乏しいこと。もう一つは、システム構築に当たり海外の OSS を採用することに対するサポート面の不安があげられる。

また、ワークフローシステム運用基盤に必要な機能やインタフェースの標準化は業界団体の WfMC がワークフロー参照モデル という形で、大きく５つのインタフェースを策定している。標準策定状況はインタフェースごとに詳細度が異なっている状況にあるが、OSS 及び商用製品を調査した結果、WfMC 標準仕様へ対応しているものは数少なく、その対応度もまちまちであることが分かった。しかし、ワークフローシステム運用基盤を選択する上で、WfMC 標準に対する対応の有無は、相互運用性と機能の基本水準を保証するという観点から、今後とも重要視されると考えている。

2 開発の目的

2.1 現状の問題点

企業における「既存システムを連携させ IT 資産の高度活用を図りたい」という要求に対する、現状のワークフローシステム運用基盤の問題点は、以下の４点と考えている。

1) 業務間連携を意識した商用製品は価格が高い。
2) 比較的大規模なシステムを前提とする等、稼働環境が限定されている。
3) 他企業とのシステム間あるいは将来導入するシステムとの相互運用性に不安がある。
4) OSS によるものは、企業によるサポートがコミットされていない。

2.2 問題解決策

本プロジェクトは、以下の特徴を持つワークフローシステム運用基盤を開発することにより、これらの問題を解決する。その結果、日本におけるワークフローシステム運用基盤の普及、しいては中堅・中小企業における経営への IT 活用の促進に寄与するものである。

1) OSS によるアプローチ

OSS のワークフローシステム運用基盤を提供することで、システム構築コストの引き下げにより、中堅・中小企業に対するシステム開発需要の拡大を目指す。

2) システムの稼働環境・規模を選ばない

JavaVM の上に構築するため、既存システムの OS 等の稼働環境に依存しない（制御対象は、Windows システムあるいは Linux システム。EJB の有無は、いずれも可能）。また、小規模なシステムから複数の商用 AP サーバ上で多数の EJB コンポーネント を利用するような大規模システムまで、既存の様々な規模の業務システムを連携させることや、本運用基盤上に様々な適用業務システムを開発することができる。

3) WfMC 標準への準拠による相互運用性の確保

WfMC 標準仕様に準拠し、ワークフローシステム運用基盤に最低限必要とされる機能を保証する。なお、ワークフロー定義仕様である WfMC のインタフェース 1 については、昨今の趨勢にあわせ、WPDL ではなく XML 形式である XPDL を採用することとする。また、標準仕様に対する対応表を作成し、対応機能を明確化する。

4) 企業によるサポート

アルゴ２１は、ワークフローシステム運用基盤を適用した受託のシステム構築では、ワークフローシステム運用基盤を含むシステム全体を一括サポートする。また、他社より製品への組み込み等のためにサポート依頼があった場合に備え、別途サポート付き商用ライセンスも用意する。

3　本システムの活用イメージ

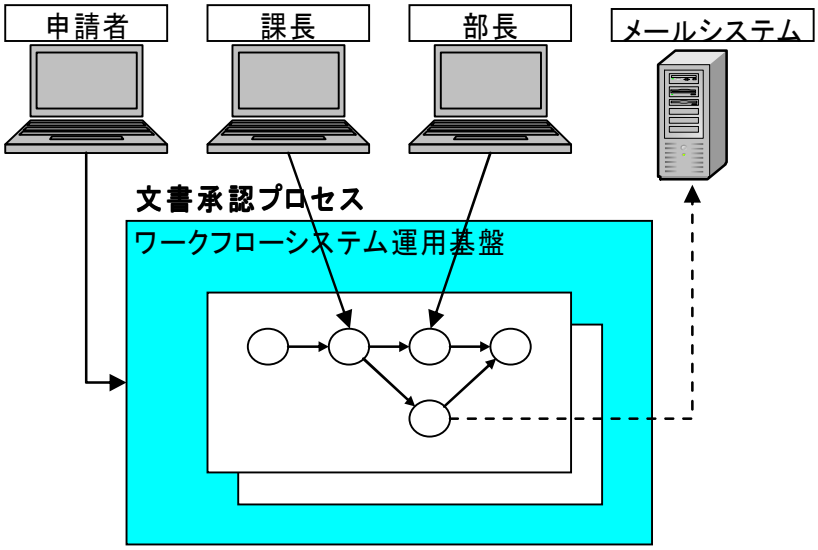
ワークフローシステム運用基盤は、ビジネスの流れを管理するための仕組みを備えている。この特徴からさまざまな形態のシステムに応用が可能であり、一般的な稟議書承認システムから、既存業務システム間の統合による新たな企業システムの構築、企業間の取引とバックエンドの業務システムとの結合等を、比較的少ない開発コストで実現することが可能である。

以下に、活用イメージとして４つの例を示す。

(1)　文書承認のワークフローエンジンとして

一般的な稟議書承認システムや文書回覧システム、経費精算システムのようなシステムの構築に、ワークフローシステム運用基盤を活用する。

申請用と承認用のアプリケーションを作成することにより、文書承認等のフロー定義とフローに沿った制御処理をワークフローシステム運用基盤で管理・制御することができる。

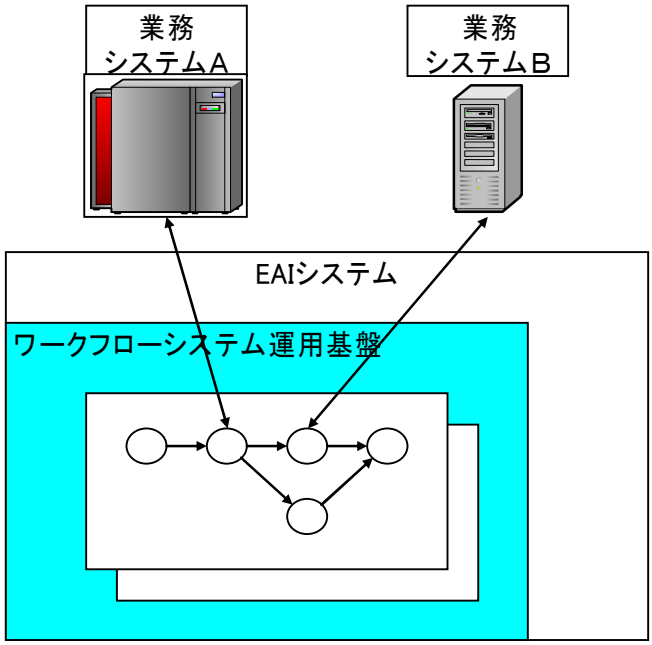


(2)　EAI ツールの連携エンジンとして

EAI システムのコア機能を実現するためのミドルウェアとしてワークフローシステム運用基盤を活用する。

個別に構築されたシステムの連携には、相互のデータ受渡し用にデータ変換プログラムを開発する必要がある。しかし直接的に複数システムを連携させてしまうと接続部分の開発工数が増加し、個別システムの変更やリプレースに対して、連携プログラムにも改修開発が発生する。EAI システムでは、データ変換によって業務システム間

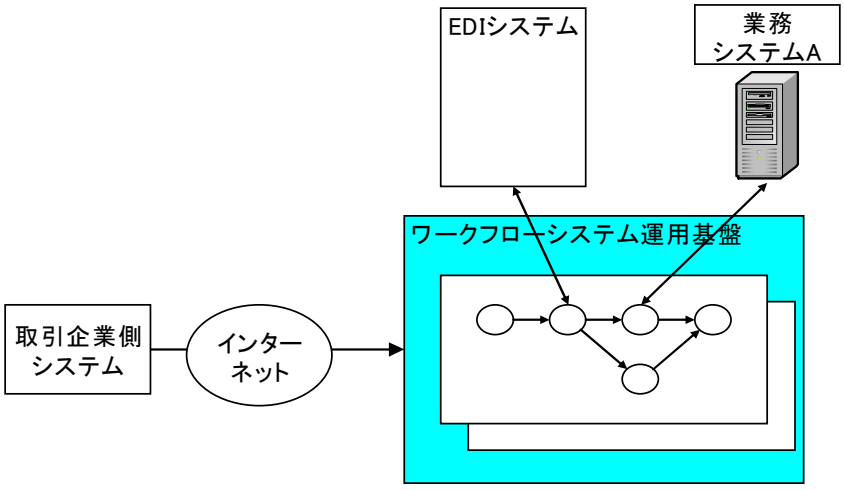
を結合することで、これらの問題を解決するが、その処理の流れをワークフローシステム運用基盤で管理・制御する。



(3) EDI ツールとバックエンドシステムの連携エンジンとして

他企業のシステムから既存の業務システムにデータを引渡す等のシステム統合に際し、EDI システムと業務システムの連携に、ワークフローシステム運用基盤を活用する。

EDI システムが受信したデータを変換した後、データの種類に応じて既存システムと連携するが、この処理の流れをワークフローシステム運用基盤で管理・制御する。

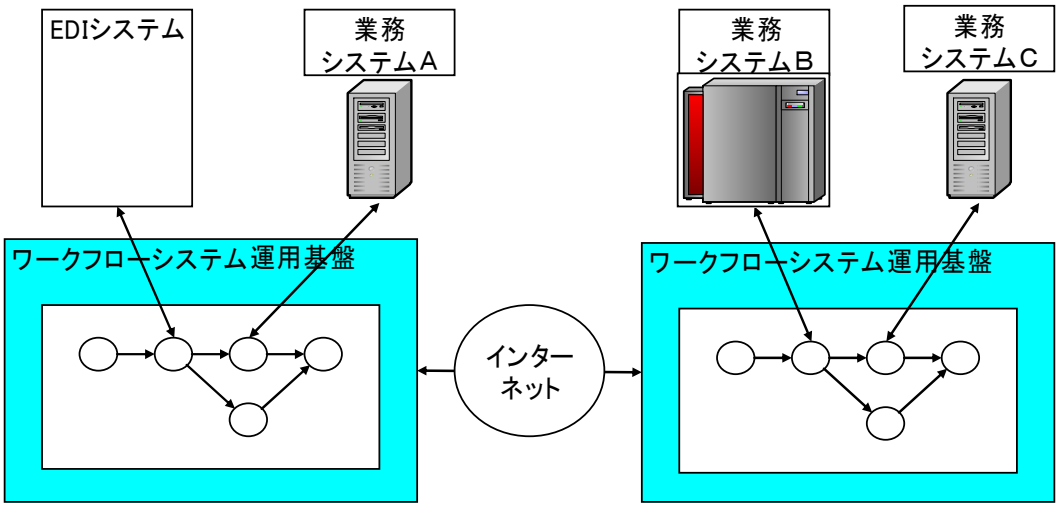


(4) 総合的な企業間システム連携のエンジンとして

総合的な BtoB(Business to Business)システム構築に当たって、ワークフローシス

テム運用基盤を活用する。

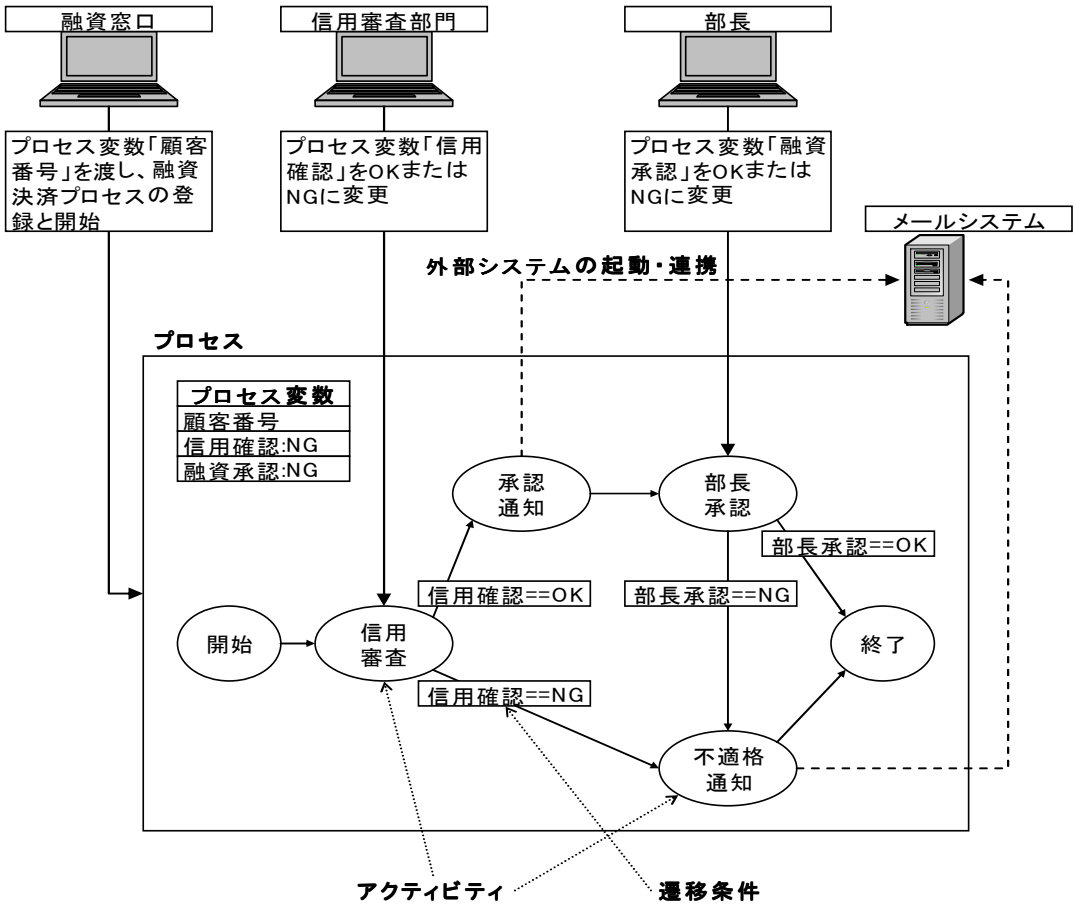
双方の企業間で同じワークフローシステム運用基盤を採用することにより、企業間にまたがる一連の処理の流れを一つのワークフローとして、複数のワークフローシステム運用基盤上で連携・制御する。



4　動作イメージ

本システムの動作イメージを以下の図に示す。ワークフローシステム運用基盤は、プロセスの雛形となるワークフロー定義の作成を支援し、クライアントによるプロセス変数の更新等のプログラミングインタフェースも提供する。また、プロセス内のアクティビティを管理し、完了後の遷移条件を判定することで、次に動作するアクティビティを決定してフローを進行させる。さらに、人手を介さないアクティビティを通じて、外部システムとの連携を実現するためのインタフェースを提供する。

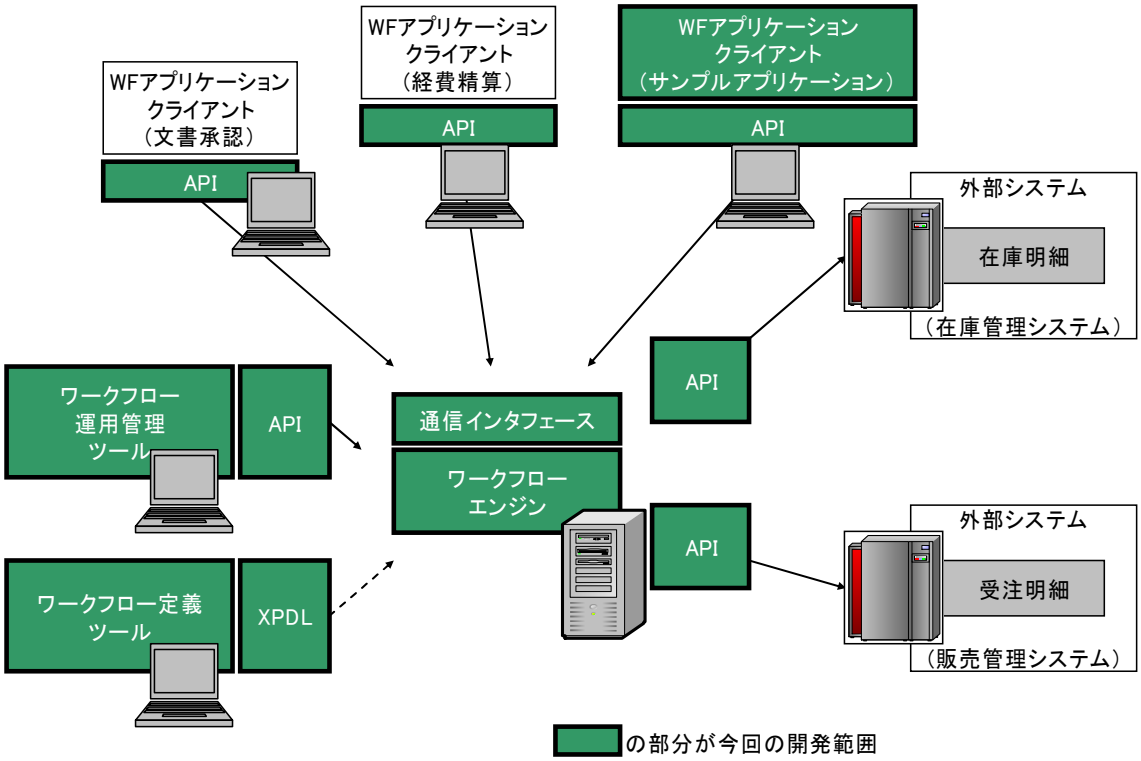
以下の図のように、プロセス変数というデータを介して、各アクティビティ間を緩やかに結合するという特徴から、既存の業務システムの統合や異なる部門や企業間の連携等を容易にし、比較的低コストで実現可能である。なお、ワークフローシステム運用基盤を使わずにこのようなシステム間連携を実現しようとすると、改修作業が各システムの様々な箇所に波及し、大規模なシステム開発になることが予想される。



5　システム構成

5.1　システム全体概要図および実現範囲

本システムでは、ミドルウェアとしてのワークフローエンジンとワークフロー定義や運用管理のための支援ツールを開発する。設計方針は、WfMC 標準仕様に準拠し、さらに軽量化を考慮することとする。また、既存の OSS ワークフローシステムにはない特徴として、複数のエンジン間の連携機能を提供する。



本プロジェクトでは今回、以下の 6 つの機能について開発を行う。

(1)　ミドルウェアとしてのワークフローエンジン

ワークフロー定義ファイルおよび、各種 API を通じて得た情報に基づき、アクティビティ及びプロセスの完了状態を管理して、プロセス変数と遷移条件によりフローの進行を制御する。また、外部システムとの連携のための API を提供し、アプリケーションの実行を管理する。

(2)　WfMC 標準に準拠したワークフローAPI

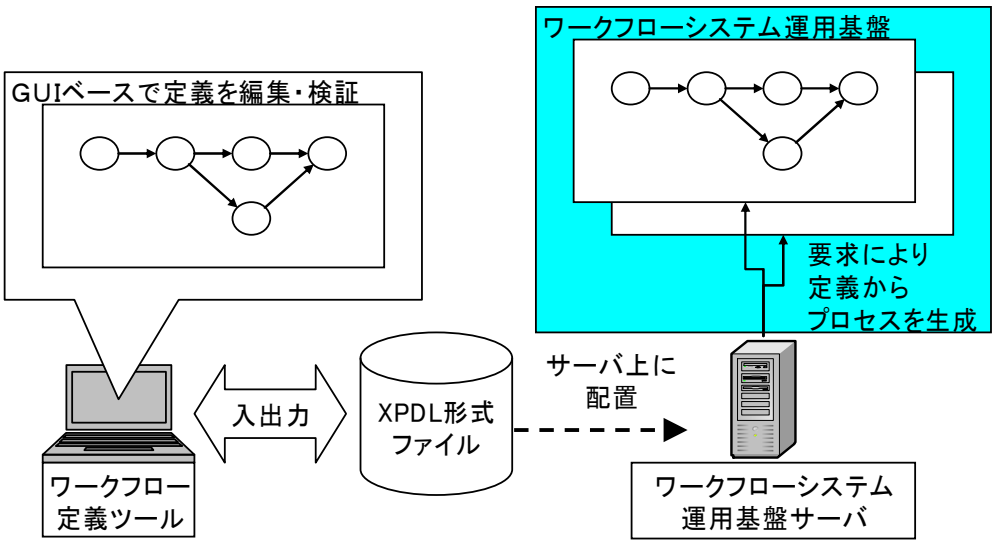
ワークフローエンジンが提供する API。 API の分類として、ワークフローシステム運用基盤の機能を利用するクライアントアプリケーション用、外部システムとのアプリケーション連携用、運用管理用、ワークフローエンジン間の連携用がある。WfMC 標準の各インタフェースは C 言語を想定した構成となっているため、Java 言語によ

る本実装においては標準インタフェース仕様に沿った機能レベルの対応を行うこととし、API と WfMC 標準インタフェースとの対応表を用意する。

(3) ワークフロー定義ツール

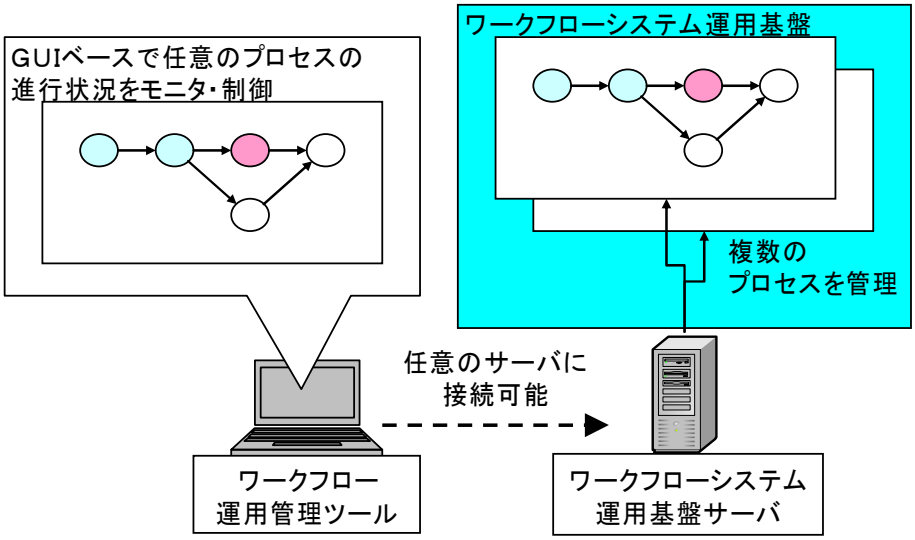
GUI によりワークフローを定義するツール。定義を構成する要素には、アプリケーション、プロセス変数、ワークフロー参加者 、プロセス、アクティビティ、遷移があり、これらの定義作業はマウスを使用して視覚的に行うことができる。

ワークフローの定義は検証を行った後、結果を XPDL 形式で保存する。



(4) ワークフロー運用管理ツール

GUI によりワークフローの実行状況を管理するツール。運用中の任意のワークフローエンジンに接続し、アクティビティの進行状況のモニタリングや、プロセス変数の編集と遷移条件の再評価を行うことができる。



(5) 外部との通信インタフェース

各種 API を RMI 、Web サービス、Servlet から使用可能とするプロトコル変換インタフェース。

(6) サンプルアプリケーション

ワークフローシステム運用基盤の機能を評価、検証するためのサンプルアプリケーション。申請・承認アプリケーションを模した小規模なプログラムとする。

5.2 ハードウェア要件

(1) ワークフローエンジン

CPU	Pentium4　2.4GHz 相当
メモリ	512MB 以上
HDD	40GB 以上

(2) ワークフロークライアント

CPU	Pentium4　1.2GHz 相当
メモリ	256MB 以上
HDD	10GB 以上

5.3 ソフトウェア要件

(1) ワークフローエンジン

OS	Redhat Enterprise Linux ES 2.1
Java VM	Sun JRE 1,4
Servlet エンジン	Jakarta Tomcat 5.0
SOAP エンジン	Apache Axis 1.2

データベースエンジン	MySQL 4.1
メッセージキュー	ActiveMQ 2.0
XML パーサ	XMLBeans 1.0

(2) ワークフロークライアント

OS	Redhat Professional Workstation 2.1 Windows XP SP2
Java VM	Sun JRE 1.4
SOAP クライアント	Apache Axis 1.2
XML パーサ	XMLBeans 1.0

5.4 ネットワーク要件

「ワークフローシステム運用基盤」周囲のネットワークについて、主な要件を次に示す。

(1) 機器間接続

ワークフローエンジンとワークフロークライアント間のネットワークは、LAN とそれらにかかわるルータ、ハブなどで構成する。ピーク時のアクセス量を想定して適切な機器を設置する。ハブは必要十分なポート数を有し、高い信頼性の機器を用意する。

(2) インターネット接続

ワークフローエンジンとワークフロークライアント間のネットワークとしてインターネットを利用することを想定するが、今回はセキュリティに対しては開発課題としない。また、検証にあたっては LAN を使用して動作確認をとるものとする。

(3) 使用プロトコルについて

インターネットで標準的に使用されているプロトコルである TCP/IP を使用する。TCP/IP 上では、送受信のプロトコルとして HTTP および IIOP を使用する。また、

5.5 配置形態

配置形態としては、ローカル VM およびサーブレットコンテナの二つを想定する。いずれも、クラスライブラリは jar ファイルで供給し、設定ファイルはクラスパス上の、awfconfig/の下にあるものとする。サーブレットの場合はこれは一意で決定するが、ローカル VM の場合は、起動シェルスクリプト内で、環境変数$NAUTICA_HOME 配下を特定するものとする。

6　検収条件

6.1　機能的検収条件

(1) ワークフローエンジン

1) 処理要求管理

サンプルアプリケーションから同時に複数の API を呼び出し、要求が順次処理されることを確認する。ワークフロー運用管理ツールを使用して確認を行う。

2) 状態遷移管理

テスト用のワークフロー定義を準備し、ワークフロー定義に指定された条件式と、プロセス変数の値に従ったアクティビティ遷移が行われることを確認する。ワークフロー運用管理ツールを使用して次の観点で確認する。

- ワークフロー参加者情報（権限設定の確認）
- プロセス変数、条件式（論理演算子、比較演算子の組み合わせを確認）

3) アプリケーション実行制御

テスト用のワークフロー定義を作成し、アプリケーションの起動と完了後のアクティビティ遷移について、ワークフロー運用管理ツールを使用して確認を行う。例外発生時の再実行については、実行ログとワークフロー運用管理ツールで確認を行う。

4) XPDL 定義ファイル解析機能

テスト用のワークフロー定義を準備し、ワークフロー運用管理ツールにより定義通りのプロセス関連情報の表示が行えることを確認する。

(2) ワークフローAPI

1) 運用管理用 API

ワークフロー運用管理ツールは全ての運用管理用 API を使用するため、ワークフロー運用管理ツールの機能確認で代替する。

2) クライアント用 API

サンプルアプリケーションはクライアント用 API を使用するため、サンプルアプリケーションの実行により機能確認を行う。

- プロセス関連情報取得、設定
- アクティビティ関連情報取得、設定

- 作業項目関連情報取得、設定

3) ワークフローエンジン間連携用 API

テストプログラムにより、ワークフローエンジン間連携用 API の機能を確認する。

- プロセスの制御、情報取得、変更

4) アプリケーション連携用 API

サンプルアプリケーションはアプリケーション連携用 API を使用するため、サンプルアプリケーションの実行により機能確認を行う。

- アプリケーションの起動、終了、情報取得

(3) ワークフロー定義ツール

ワークフロー定義ツールで定義した内容がワークフローエンジンに引き継がれ、ワークフロー運用管理ツールで定義通り表示されることで確認する。

ワークフロー定義時、必須項目チェックと印刷機能の確認を行う。

(4) ワークフロー運用管理ツール

ワークフローエンジンが制御しているプロセス、アクティビティの稼動情報を表示、変更、削除できることを確認する。

ネットワーク上の任意のサーバで稼動するワークフローエンジンに接続できることを確認する。

(5) 通信インタフェース

1) RMI over IIOP

ワークフロー運用管理ツールは RMI over IIOP を使用して運用管理用 API を利用している。ワークフロー運用管理ツールを使用して機能確認を行う。

2) Web サービス

サンプルアプリケーションは Web サービスを使用してクライアント用 API を利用している。サンプルアプリケーションを使用して機能確認を行う。

3) Servlet

提供 Servlet に HTML 変換部分を追加し、Web ブラウザから Servlet を経由して運用管理用 API が利用できることを確認する。

4) サンプルアプリケーション

サンプルアプリケーションの機能動作は、ワークフロー運用管理ツールを使用してプロセスの進行状況を照会することで確認を行う。

6.2 性能的検収条件

(1) ワークフローエンジン

- ワークフロー定義を、100 定義以上管理可能であること。
- 処理要求発生後、アクティビティ状態遷移は 10 秒以内であること。

(2) ワークフロー定義ツール

- 定義作業時のマウス操作の反応速度が 3 秒以下であること。

(3) ワークフロー運用管理ツール

- プロセス情報に対する参照、更新要求の応答が最大 8 秒以下であること。

6.3 普及活動等に向けての行動および具備すべき条件

- アルゴ２１のプレスリリースとして本プロジェクトの立ち上げをアナウンスする。
- SourceForge.jp にプロジェクトホームページを開設し、開発成果物を公開する。
- ワークフローシステム運用基盤の取扱説明書を作成し、ホームページに公開する。

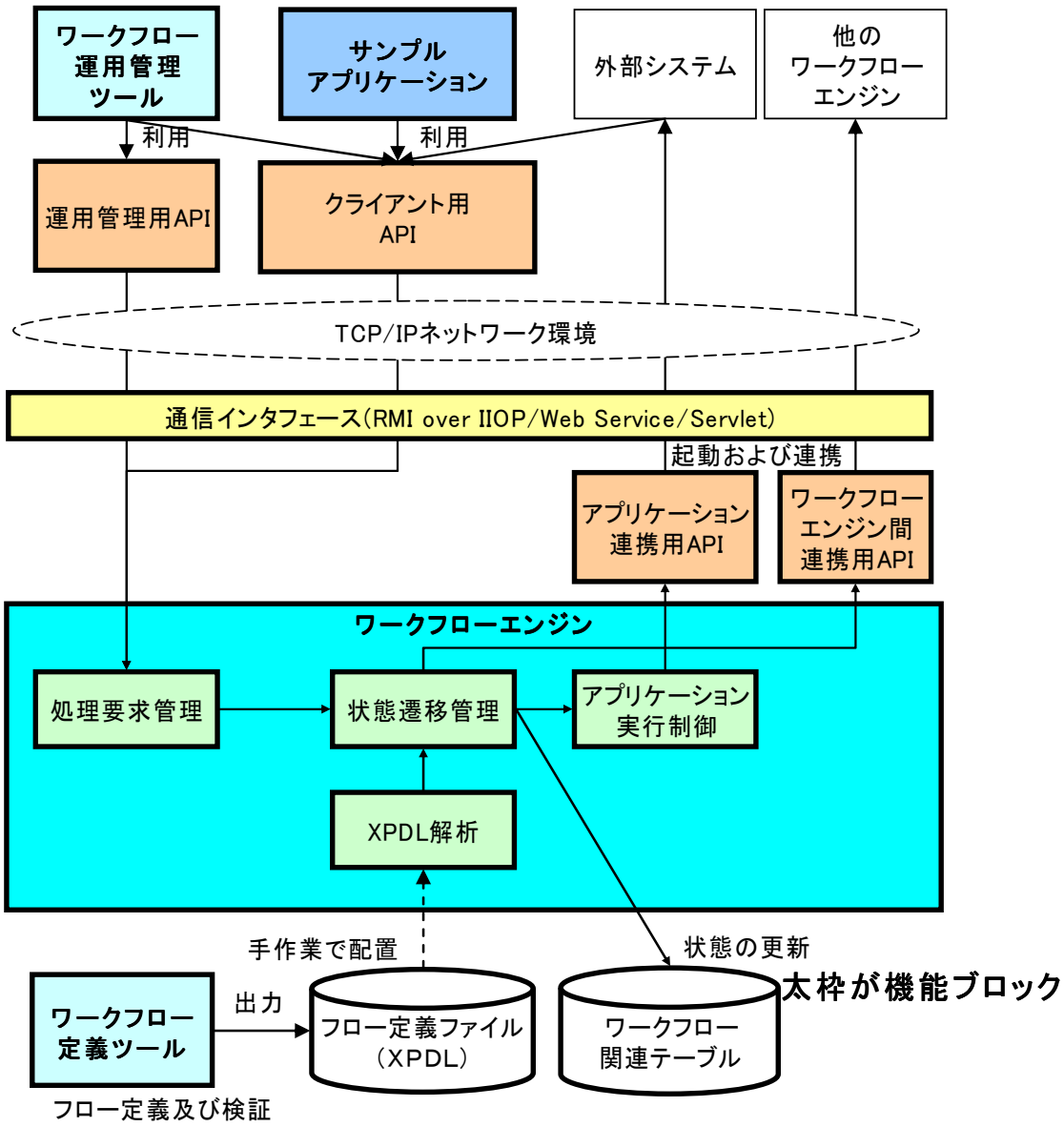
7　ライセンス

ソフトウェア成果物のライセンスは以下のとおりとする。ただし、商用製品への組み込み等でアルゴ２１によるサポートを必要とする企業に対しては、別途商用ライセンスを準備する予定である。

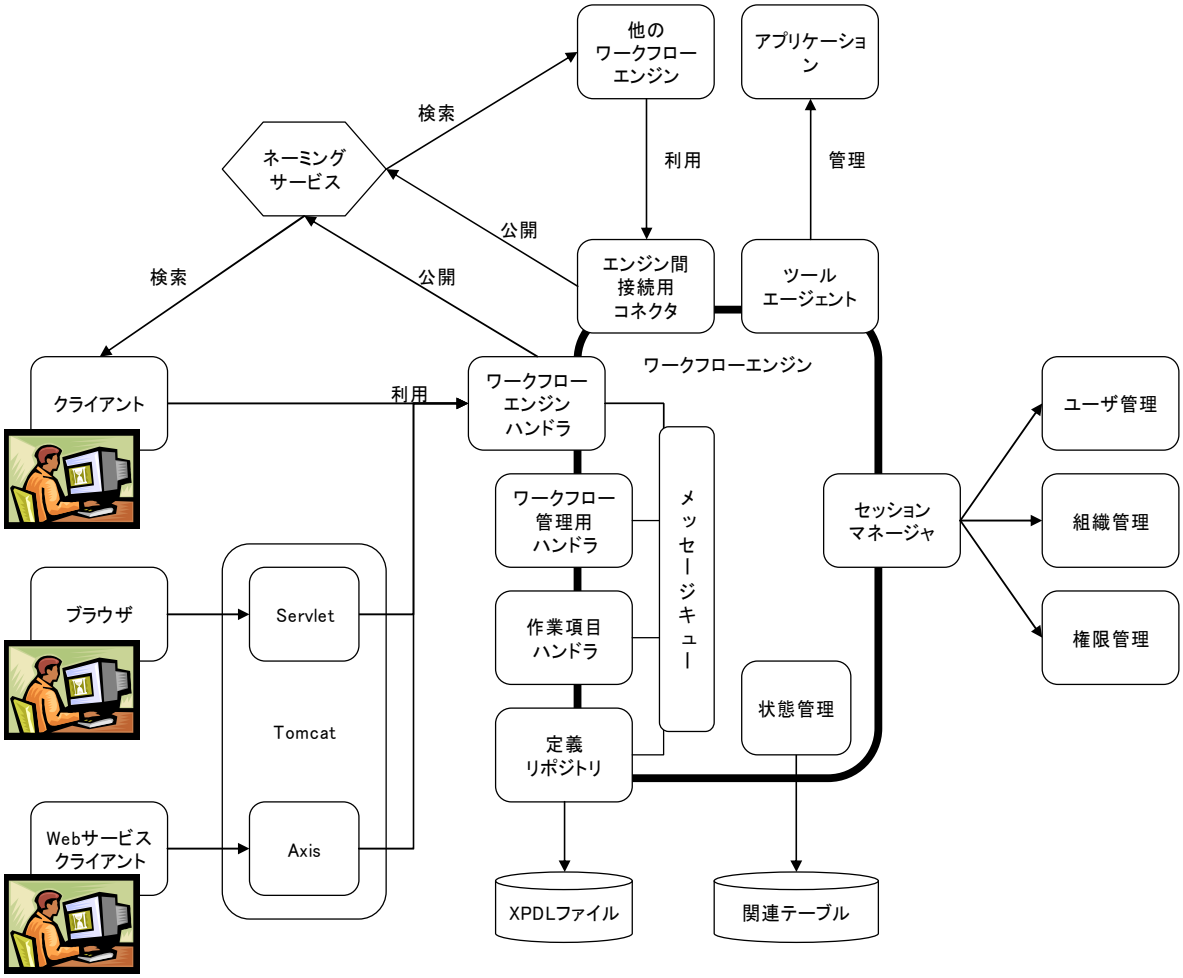
- ワークフローエンジン　：　GNU Lesser General Public License
- ワークフローＡＰＩ　：　GNU Lesser General Public License
- 通信インタフェース　：　GNU Lesser General Public License
- ワークフロー定義ツール　：　GNU General Public License
- ワークフロー運用管理ツール：　GNU General Public License
- サンプルアプリケーション　：　GNU General Public License

II　機能仕様編

1　機能構成図


ワークフロー
運用管理
ツール
サンプル
アプリケーション
外部システム
他の
ワークフロー
エンジン
利用
利用
運用管理用API
クライアント用
API
TCP/IPネットワーク環境
通信インタフェース(RMI over IIOP/Web Service/Servlet)
起動および連携
アプリケーション
連携用API
ワークフロー
エンジン間
連携用API
ワークフローエンジン
処理要求管理
状態遷移管理
アプリケーション
実行制御
XPDL解析
手作業で配置
状態の更新
ワークフロー
定義ツール
出力
フロー定義ファイル
（XPDL）
ワークフロー
関連テーブル
太枠が機能ブロック
フロー定義及び検証

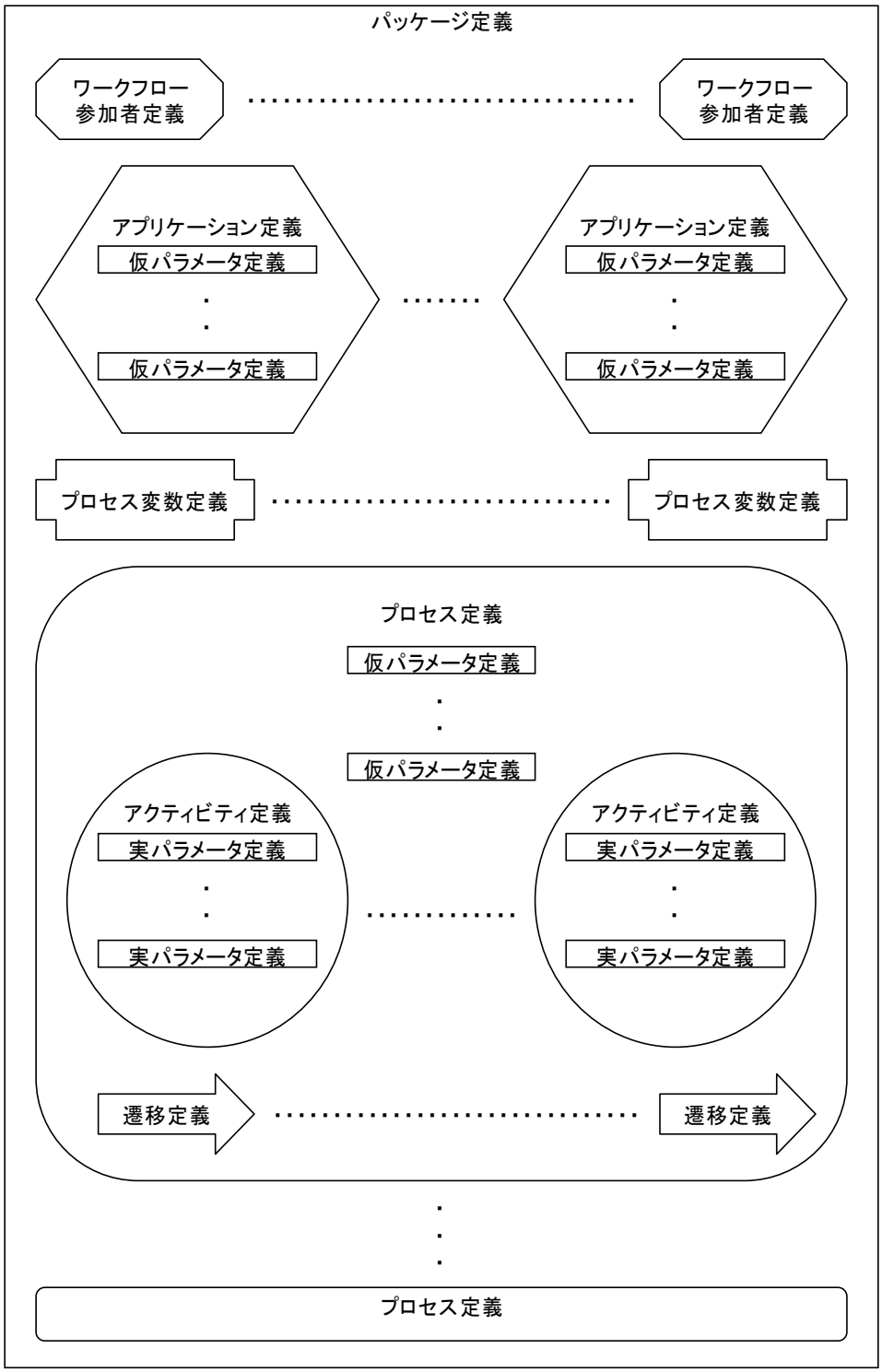
2　モジュール構成


他の
ワークフロー
エンジン
アプリケーション
検索
利用
管理
ネーミング
サービス
公開
検索
公開
エンジン間
接続用
コネクタ
ツール
エージェント
ワークフローエンジン
クライアント
利用
ワークフロー
エンジン
ハンドラ
ユーザ管理
ワークフロー
管理用
ハンドラ
メッセージキュー
セッション
マネージャ
組織管理
ブラウザ
Servlet
作業項目
ハンドラ
権限管理
Tomcat
状態管理
定義
リポジトリ
Webサービス
クライアント
Axis
XPDLファイル
関連テーブル

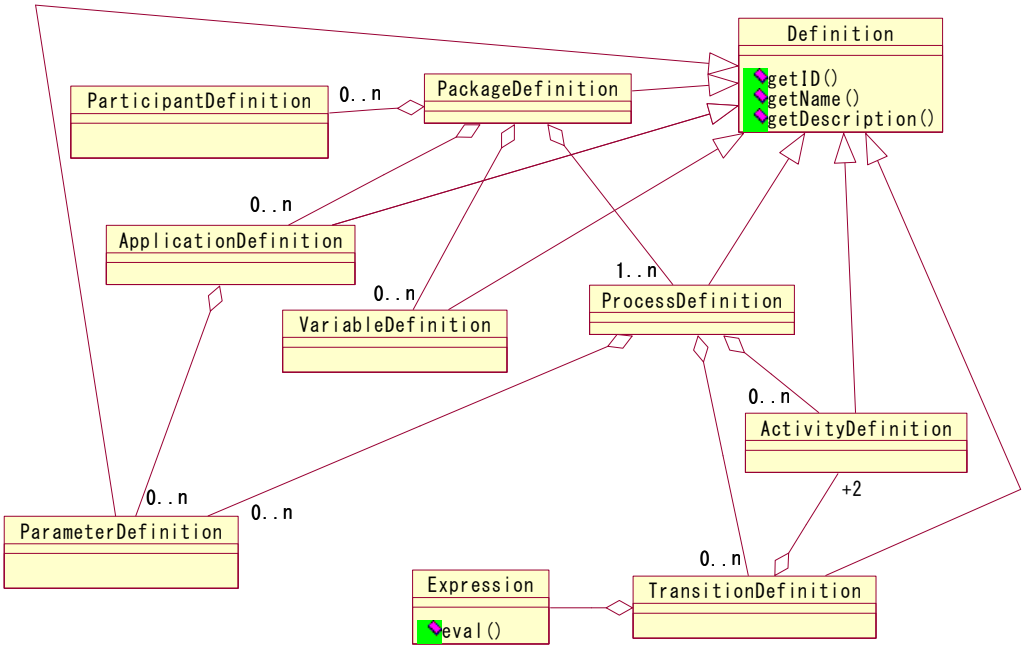
3　基本定義要素

3.1　ワークフロー定義

3.1.1　XPDL の定義構成



3.1.2　クラス図



名　　称	内　　容
Definition	すべての定義要素の基底インタフェース。関連する他の定義要素を一つに纏める要素である。
PackageDefinition	ワークフロー定義パッケージを表す定義要素。
PaticipantDefinition	ワークフロー参加者を表す定義要素。
ApplicationDefinition	ワークフローアプリケーションを表す定義要素。パラメータ定義要素を持つ場合がある。
VariableDefinition	プロセス変数を表す定義要素。
ProcessDefinition	プロセスを表す定義要素。パラメータ定義要素を持つ場合がある。
ActivityDefinition	アクティビティ情報を表す定義要素。任意のアクティビティ定義要素は、いずれかのプロセスに所属している。
TransitionDefinition	遷移を表す定義要素。遷移元と遷移先のアクティビティ定義要素を持つ。遷移には、条件式が伴う場合がある。
ParameterDefinition	パラメータを表す定義要素。
Expression	すべての条件式の基底インタフェース。条件評価のために eval()メソッドを持つ。

3.2　パッケージ定義

3.2.1 概要

パッケージ定義内には、複数のワークフロー参加者定義、アプリケーション定義、プロセス変数定義、プロセス定義が含まれる。含まれるプロセス定義は、ひとつのメインプロセスの定義と、０個以上のサブプロセス定義で構成するものとする。

3.2.2 インタフェース

パッケージ定義は以下のインタフェースを持つ。

インタフェース	説　　明
java.util.List getApplications()	アプリケーション一覧を返す。
String getAuthor()	定義者を返す。
String getCreatedDate()	作成日を返す。
String getDescription()	説明を返す。
String getID()	ID を返す。
String getName()	名称を返す。
java.util.List getParticipants()	ワークフロー参加者一覧を返す。
java.util.List getProcesses()	プロセス一覧を返す。
java.util.List getVariables()	プロセス変数一覧を返す。
String getVersion()	バージョンを返す。
String getXPDLVersion()	XPDL バージョンを返す。

3.2.3 XML スキーマ定義と XML 表現

パッケージ定義の XML スキーマの概要は、以下のようなものである。詳細なスキーマ定義は、III 入出力データ編を参照のこと。

```
<xsd:element name="Package">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="xpdl:PackageHeader"/>
      <xsd:element ref="xpdl:RedefinableHeader" minOccurs="0"/>
      <xsd:element ref="xpdl:ConformanceClass" minOccurs="0"/>
      <xsd:element ref="xpdl:Script" minOccurs="0"/>
      <xsd:element ref="xpdl:ExternalPackages" minOccurs="0"/>
      <xsd:element ref="xpdl:TypeDeclarations" minOccurs="0"/>
      <xsd:element ref="xpdl:Participants" minOccurs="0"/>
      <xsd:element ref="xpdl:Applications" minOccurs="0"/>
      <xsd:element ref="xpdl:DataFields" minOccurs="0"/>
      <xsd:element ref="xpdl:WorkflowProcesses" minOccurs="0"/>
      <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
    <xsd:attribute name="Name" type="xsd:string"/>
  </xsd:complexType>
</xsd:element>
```

また、このスキーマにより定義されるパッケージ定義の例を以下に示す。

```
<xpdl:Package Name="交通費" Id="PK110717053304600001">
   <xpdl:PackageHeader>
      <xpdl:XPDLVersion>1.0</xpdl:XPDLVersion>
      <xpdl:Vendor>ARGO 21, Corporation</xpdl:Vendor>
      <xpdl:Created>2005/1/31</xpdl:Created>
      <xpdl:Description>aaaa</xpdl:Description>
   </xpdl:PackageHeader>
   <xpdl:RedefinableHeader>
      <xpdl:Author>nito</xpdl:Author>
      <xpdl:Version>1.0</xpdl:Version>
   </xpdl:RedefinableHeader>
   <xpdl:DataFields/>
   <xpdl:Participants/>
   <xpdl:Applications/>
   <xpdl:WorkflowProcesses/>
</xpdl:Package>
```

3.3 ワークフロー参加者定義

3.3.1 概要

ワークフロー参加者は、ワークフローに参加するリソースを表しており、以下の種類の定義が可能である。

- システム
- 組織単位
- ユーザ
- ロール（権限）

3.3.2 インタフェース

ワークフロー参加者定義は以下のインタフェースを持つ。

インタフェース	説　　明
String getDescription()	説明を返す。
String getID()	ID を返す。
String getName()	名称を返す。
int getType()	参加者型を返す。 **システム** **組織単位** **ユーザ** **ロール**

3.3.3 XML スキーマ定義と XML 表現

ワークフロー参加者定義の XML スキーマの概要は、以下のようなものである。詳細なスキーマ定義は、III 入出力データ編を参照のこと。

```
<xsd:element name="Participant">
```

```
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="xpdl:ParticipantType"/>
      <xsd:element ref="xpdl:Description" minOccurs="0"/>
      <xsd:element ref="xpdl:ExternalReference" minOccurs="0"/>
      <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
    <xsd:attribute name="Name" type="xsd:string"/>
  </xsd:complexType>
</xsd:element>
```

また、このスキーマにより定義されるワークフロー参加者定義の例を以下に示す。

```
<xpdl:Participant Name="部長" Id="PP110717065585900000">
   <xpdl:ParticipantType Type="ORGANIZATIONAL_UNIT"/>
   <xpdl:Description>ぶちょう</xpdl:Description>
</xpdl:Participant>
```

3.4 ワークフローアプリケーション定義

3.4.1 概要

ワークフローアプリケーションは、ワークフローから起動されるアプリケーションであり、アプリケーションに引き渡すべきパラメータの定義を含む。

3.4.2 インタフェース

ワークフローアプリケーション定義は以下のインタフェースを持つ。

インタフェース	説　　明
String getDescription()	説明を返す。
String getID()	ID を返す。
String getName()	名称を返す。
java.util.List getParameters()	パラメータ一覧を返す。

3.4.3 XML スキーマ定義と XML 表現

ワークフローアプリケーション定義の XML スキーマの概要は、以下のようなものである。詳細なスキーマ定義は、III 入出力データ編を参照のこと。

```
<xsd:element name="Application">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="xpdl:Description" minOccurs="0"/>
      <xsd:choice>
        <xsd:element ref="xpdl:FormalParameters"/>
        <xsd:element ref="xpdl:ExternalReference" minOccurs="0"/>
      </xsd:choice>
      <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
    <xsd:attribute name="Name" type="xsd:string"/>
```

```
  </xsd:complexType>
</xsd:element>
```

また、このスキーマにより定義されるワークフローアプリケーション定義の例を以下に示す。

```
<xpdl:Application Name="メール送信" Id="AP110717072134300000">
   <xpdl:Description>メール</xpdl:Description>
   <xpdl:FormalParameters/>
</xpdl:Application>
```

3.5 プロセス変数定義

3.5.1 概要

プロセス変数は、その値が判定されることにより、ワークフローの進行を制御する変数を定義するものである。変数の種類としては、文字列型、数値型、真理値型を提供するものとする。

3.5.2 インタフェース

プロセス変数定義は以下のインタフェースを持つ。

インタフェース	説　　明
String getDescription()	説明を返す。
String getID()	ID を返す。
String getInitialValue()	初期値を返す。
String getName()	名称を返す。
int getType()	型を返す。 文字列型 数値型 真理値型

3.5.3 XML スキーマ定義と XML 表現

プロセス変数定義の XML スキーマの概要は、以下のようなものである。詳細なスキーマ定義は、III 入出力データ編を参照のこと。

```
<xsd:element name="DataField">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="xpdl:DataType"/>
      <xsd:element ref="xpdl:InitialValue" minOccurs="0"/>
      <xsd:element ref="xpdl:Length" minOccurs="0"/>
      <xsd:element ref="xpdl:Description" minOccurs="0"/>
      <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
    <xsd:attribute name="Name" type="xsd:string"/>
    <xsd:attribute name="IsArray" default="FALSE">
```

```
    <xsd:simpleType>
      <xsd:restriction base="xsd:NMTOKEN">
        <xsd:enumeration value="TRUE"/>
        <xsd:enumeration value="FALSE"/>
      </xsd:restriction>
    </xsd:simpleType>
    </xsd:attribute>
  </xsd:complexType>
</xsd:element>
```

また、このスキーマにより定義されるプロセス変数定義の例を以下に示す。

```
<xpdl:DataField Name="あて先" Id="VR110717083765600000">
   <xpdl:DataType>
      <xpdl:BasicType Type="STRING"/>
   </xpdl:DataType>
   <xpdl:InitialValue/>
   <xpdl:Description>申請者のメールアドレス</xpdl:Description>
</xpdl:DataField>
```

3.6 プロセス定義

3.6.1 概要

プロセス定義は、ワークフローを構成する一連の処理の流れを定義するものである。プロセスは、一つの処理を表すアクティビティと、アクティビティ間を結合する遷移から構成されており、プロセス定義にはそれらを内部に含む形で定義される。

本ワークフローシステム運用基盤では、パッケージ内にひとつのメインプロセスと、それから起動される０個以上のサブプロセスという構成を採用する。メインプロセスとサブプロセスは､アクセスレベル属性(PUBLIC/PRIVATE)によって識別するものとする。

3.6.2 インタフェース

プロセス定義は以下のインタフェースを持つ。

インタフェース	説　　明
int getAccessLevel()	アクセスレベルを返す。 PUBLIC：メインプロセスとする PRIVATE：サブプロセスとする
java.util.List getActivities()	アクティビティ一覧を返す。
String getDescription()	説明を返す。
int getDurationUnit()	期間単位を返す。 年 月 日 時 分 秒

String getID()	ID を返す。
int getLimit()	実行期限を返す。
int getLogicalHeight()	論理座標高さを返す。
int getLogicalWidth()	論理座標幅を返す。
String getName()	名称を返す。
java.util.List getParameters()	パラメーター一覧を返す。
java.util.List getTransitions()	遷移一覧を返す。

3.6.3 XML スキーマ定義と XML 表現

プロセス定義の XML スキーマの概要は、以下のようなものである。詳細なスキーマ定義は、III 入出力データ編を参照のこと。

```
<xsd:element name="WorkflowProcess">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="xpdl:ProcessHeader"/>
      <xsd:element ref="xpdl:RedefinableHeader" minOccurs="0"/>
      <xsd:element ref="xpdl:FormalParameters" minOccurs="0"/>
      <xsd:element ref="xpdl:DataFields" minOccurs="0"/>
      <xsd:element ref="xpdl:Participants" minOccurs="0"/>
      <xsd:element ref="xpdl:Applications" minOccurs="0"/>
      <xsd:element ref="xpdl:ActivitySets" minOccurs="0"/>
      <xsd:element ref="xpdl:Activities" minOccurs="0"/>
      <xsd:element ref="xpdl:Transitions" minOccurs="0"/>
      <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
    <xsd:attribute name="Name" type="xsd:string"/>
    <xsd:attribute name="AccessLevel">
      <xsd:simpleType>
        <xsd:restriction base="xsd:NMTOKEN">
          <xsd:enumeration value="PUBLIC"/>
          <xsd:enumeration value="PRIVATE"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:attribute>
  </xsd:complexType>
</xsd:element>
```

また、このスキーマにより定義されるプロセス定義の例を以下に示す。

```
<xpdl:WorkflowProcess Id="--UNINITIALIZED--" Name="開始" AccessLevel="PUBLIC">
   <xpdl:ProcessHeader DurationUnit="D">
      <xpdl:Limit>0</xpdl:Limit>
   </xpdl:ProcessHeader>
   <xpdl:Activities/>
   <xpdl:Transitions/>
   <xpdl:ExtendedAttributes>
      <xpdl:ExtendedAttribute Name="LOGICAL_WIDTH" Value="10"/>
      <xpdl:ExtendedAttribute Name="LOGICAL_HEIGHT" Value="10"/>
   </xpdl:ExtendedAttributes>
</xpdl:WorkflowProcess>
```

3.7 アクティビティ定義

3.7.1 概要

アクティビティ定義は、ワークフローにおけるひとつの処理に対応する概念であるアクティビティの定義を行うものである。なお、本ワークフローシステム運用基盤では、以下の種類のアクティビティを提供する。

種類	説　　明
開始アクティビティ	プロセスの開始を表す。
対話型アクティビティ	クライアントプログラムからの制御を待つ。
自動実行アクティビティ	ワークフローアプリケーションを起動する。
サブプロセス起動アクティビティ	サブプロセスの起動を行う。
並行処理開始アクティビティ	並行処理ブロックの開始を表す。条件に合致した遷移をすべて動作させる。
並行処理終了アクティビティ	並行処理ブロックの終了を表す。並行処理開始で動作した遷移をすべて待ち合わせてから終了する。
終了アクティビティ	プロセスの終了を表す。

3.7.2 インタフェース

アクティビティ定義は以下のインタフェースを持つ。

インタフェース	説　　明
String getAffiliateBID()	所属ブロック ID を返す。 並行処理開始アクティビティ、並行処理終了アクティビティ以外で有効。値としては、従属している並行処理開始アクティビティのブロック ID と同じ値を保持している。
String getApplicationID()	アプリケーション ID を返す。自動実行で有効。
java.util.List getApplicationParameters()	アプリケーションパラメーター一覧を返す。自動実行で有効。
String getBlockID()	ブロック ID を返す。並行処理開始アクティビティ、並行処理終了

	アクティビティで有効。
String getDescription()	説明を返す。
int getExecution()	サブプロセス実行時の同期状態を返す。 サブプロセス起動で有効。 同期(SYNCHR) 非同期(ASYNCHR)
String getID()	ID を返す。
int getLimit()	実行期限を返す。
int getLogicalX()	論理 X 座標を返す。
int getLogicalY()	論理 Y 座標を返す。
String getName()	アクティビティ名称を返す。
String getParentBID()	親ブロック ID を返す。
String getPerformerID()	実行許可者 ID を返す。ワークフロー参加者 ID が入る。
String getSubprocessID()	サブプロセス ID を返す。 サブプロセス起動で有効。
java.util.List getSubprocessParameters()	サブプロセスパラメーター一覧を返す。 サブプロセス起動で有効。
String getType()	アクティビティ型を返す。 開始アクティビティ 対話型アクティビティ 自動実行アクティビティ サブプロセス起動アクティビティ 並行処理開始アクティビティ 並行処理終了アクティビティ 終了アクティビティ

3.7.3 XML スキーマ定義と XML 表現

アクティビティ定義の XML スキーマの概要は、以下のようなものである。詳細なスキーマ定義は、III 入出力データ編を参照のこと。

```
<xsd:element name="Activity">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="xpdl:Description" minOccurs="0"/>
      <xsd:element ref="xpdl:Limit" minOccurs="0"/>
      <xsd:choice>
        <xsd:element ref="xpdl:Route"/>
        <xsd:element ref="xpdl:Implementation"/>
        <xsd:element ref="xpdl:BlockActivity"/>
      </xsd:choice>
      <xsd:element ref="xpdl:Performer" minOccurs="0"/>
      <xsd:element ref="xpdl:StartMode" minOccurs="0"/>
```

```
      <xsd:element ref="xpdl:FinishMode" minOccurs="0"/>
      <xsd:element ref="xpdl:Priority" minOccurs="0"/>
      <xsd:element ref="xpdl:Deadline" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element ref="xpdl:SimulationInformation" minOccurs="0"/>
      <xsd:element ref="xpdl:Icon" minOccurs="0"/>
      <xsd:element ref="xpdl:Documentation" minOccurs="0"/>
      <xsd:element ref="xpdl:TransitionRestrictions" minOccurs="0"/>
      <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
    <xsd:attribute name="Name" type="xsd:string"/>
  </xsd:complexType>
</xsd:element>
```

また、このスキーマにより定義されるアクティビティ定義の例を以下に示す。

```
<xpdl:Activity Name="認可メール送信処理" Id="AC110717113575000000">
   <xpdl:Description>申請者にメールを送る</xpdl:Description>
   <xpdl:Limit>10</xpdl:Limit>
   <xpdl:Implementation>
      <xpdl:Tool Id="AP110717072134300000">
         <xpdl:ActualParameters>
            <xpdl:ActualParameter>VR110717083765600000</xpdl:ActualParameter>
            <xpdl:ActualParameter/>
         </xpdl:ActualParameters>
      </xpdl:Tool>
   </xpdl:Implementation>
   <xpdl:Performer>PP110717062185900000</xpdl:Performer>
   <xpdl:ExtendedAttributes>
      <xpdl:ExtendedAttribute Value="3" Name="LOGICAL_X"/>
      <xpdl:ExtendedAttribute Value="1" Name="LOGICAL_Y"/>
      <xpdl:ExtendedAttribute Value="AUTOMATIC" Name="ACTIVITY_TYPE"/>
   </xpdl:ExtendedAttributes>
</xpdl:Activity>
```

3.8 遷移定義

3.8.1 概要

遷移定義は、アクティビティ間の遷移を定義するものである。遷移には、遷移のための条件指定を持つ場合がある。

3.8.2 インタフェース

遷移定義は以下のインタフェースを持つ。

インタフェース	説　　明
Expression getCondition()	遷移条件を返す。
java.lang.String getDescription()	説明を返す。
ActivityElement getDestination()	遷移先アクティビティを返す。
java.lang.String getID()	ID を返す。
java.lang.String getName()	名称を返す。
ActivityElement getSource()	遷移元アクティビティを返す。

int getType()	遷移型を返す。 条件指定(CONDITION) 例外(EXCEPTION) その他(OTHERWISE)

3.8.3 XML スキーマ定義と XML 表現

遷移定義の XML スキーマの概要は、以下のようなものである。詳細なスキーマ定義は、III 入出力データ編を参照のこと。

```
<xsd:element name="Transition">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="xpdl:Condition" minOccurs="0"/>
      <xsd:element ref="xpdl:Description" minOccurs="0"/>
      <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
    <xsd:attribute name="From" type="xsd:NMTOKEN" use="required"/>
    <xsd:attribute name="To" type="xsd:NMTOKEN" use="required"/>
    <xsd:attribute name="Name" type="xsd:string"/>
  </xsd:complexType>
</xsd:element>
```

また、このスキーマにより定義される遷移定義の例を以下に示す。

```
<xpdl:Transition To="AC110717113975000000" From="AC110717113200000000" Name="" Id="TR110717116490600000">
   <xpdl:Condition Type="CONDITION">
      <awf:ConditionalExpression>
         <awf:RelationalExpression ComparedValue="bbbb" DataFieldRef="VR110717083765600000" Type="EQUAL"/>
      </awf:ConditionalExpression>
   </xpdl:Condition>
   <xpdl:Description/>
</xpdl:Transition>
```

3.9 パラメータ定義

3.9.1 概要

パラメータ定義は、アプリケーションまたはプロセスに渡すべきパラメータ情報を定義するものである。

3.9.2 インタフェース

パラメータ定義は以下のインタフェースを持つ。

インタフェース	説　　明
String getDescription()	説明を返す。
String getID()	ID を返す。
int getMode()	入出力モードを返す。 入力(IN) 出力(OUT) 入出力(INOUT)

String getName()	名称を返す。
int getType()	パラメータ型を返す。 文字列型 数値型 真理値型

3.9.3 XML スキーマ定義と XML 表現

パラメータ定義の XML スキーマの概要は、以下のようなものである。詳細なスキーマ定義は、III 入出力データ編を参照のこと。

```
<xsd:element name="FormalParameter">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="xpdl:DataType"/>
      <xsd:element ref="xpdl:Description" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
    <xsd:attribute name="Index" type="xsd:NMTOKEN"/>
    <xsd:attribute name="Mode" default="IN">
    <xsd:simpleType>
      <xsd:restriction base="xsd:NMTOKEN">
        <xsd:enumeration value="IN"/>
        <xsd:enumeration value="OUT"/>
        <xsd:enumeration value="INOUT"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
</xsd:complexType>
</xsd:element>
```

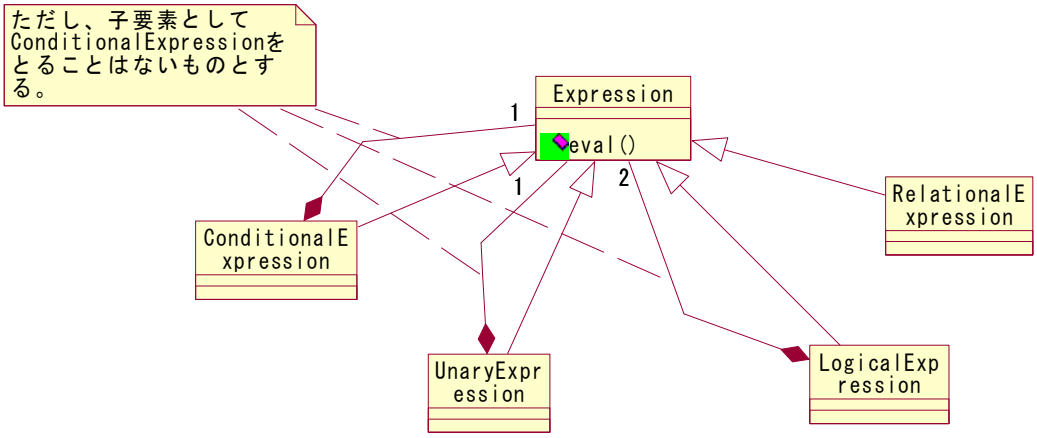
また、このスキーマにより定義されるパラメータ定義の例を以下に示す。

```
<xpdl:FormalParameter Mode="IN" Id="PM110717076168700000">
   <xpdl:DataType>
      <xpdl:BasicType Type="STRING"/>
   </xpdl:DataType>
   <xpdl:Description>あて先</xpdl:Description>
</xpdl:FormalParameter>
```

3.10 条件式

3.10.1 クラス図

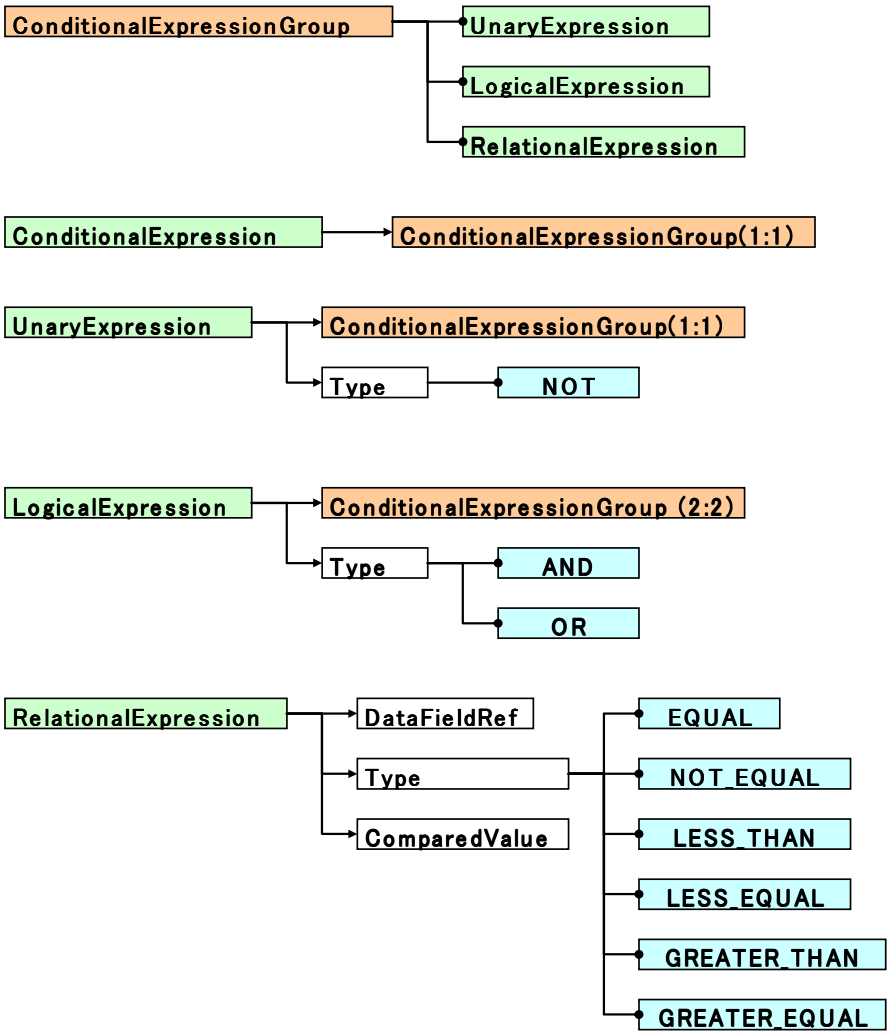
条件式は、eval()メソッドを持つ Expression から導出した４つのクラスから構成される。



名称	内容
Expression	すべての条件式の基底インタフェース。条件評価のために eval()メソッドを持つ。
ConditionalExpression	条件式を取りまとめる最上位のクラス。内部に、1 個の条件式を保持する。
UnaryExpression	単項条件式を表現するクラス。内部に、1 個の条件式を保持する。ここでは、NOT 演算のみに対応。
LogicalExpression	論理条件式を表現するクラス。内部に、2 個の条件式を保持する。ここでは、AND／OR 演算に対応。
RelationalExpression	関係条件式を表現するクラス。ここでは、文字列／数値／真理演算のみに対応。

3.10.2 XML スキーマ定義と XML 表現

条件式の XML スキーマの概要は、以下のようなものである。詳細なスキーマ定義は、III 入出力データ編を参照のこと。



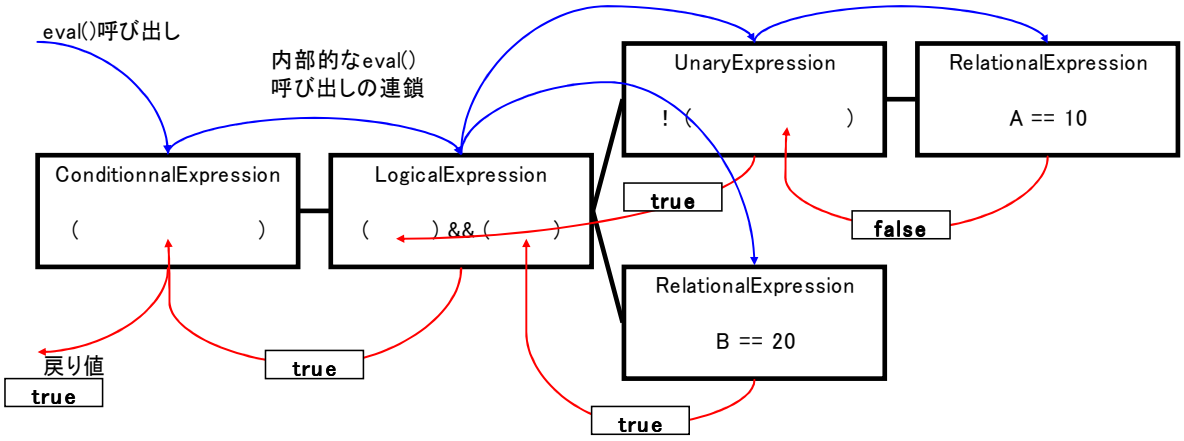
また、このスキーマにより定義される条件式の例を以下に示す。例では、((! (A == 10)) && (B == 20))という条件を定義している。

```
<ConditionDefinition>
  <LogicalDefinition Type="AND">
    <UnaryDefinition Type="NOT">
      <RelationalDefinition DataFieldRef="A" Type="EQUAL" ComparedValue="10" />
    </UnaryDefinition>
    <RelationalDefinition DataFieldRef="B" Type="EQUAL" ComparedValue="20" />
  </LogicalDefinition>
</ConditionDefinition>
```

3.10.3 条件判定の流れ

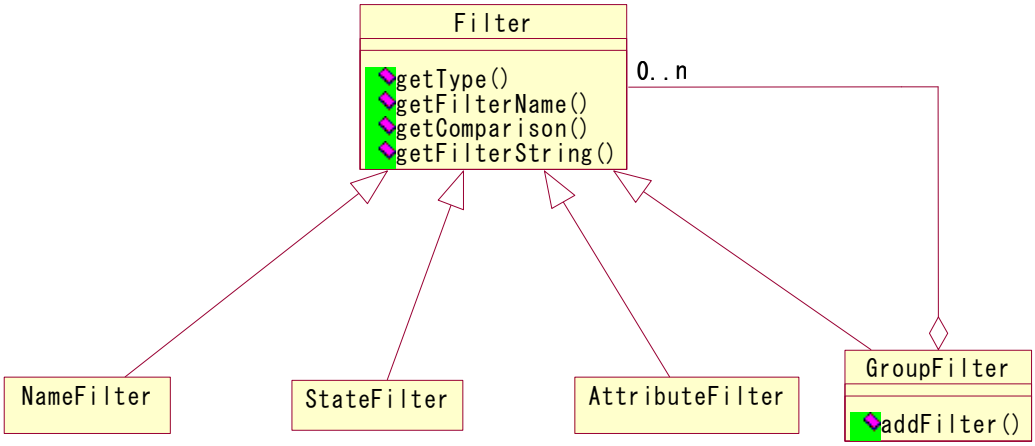
条件判定は、応答連鎖(Chain of Responsibility)パターンにより行われる。以下の例では、((! (A == 10)) && (B == 20))という条件を評価する流れを示す。基本的な流れは、保持している入れ子の条件式があれば、まず入れ子の条件式の eval()を呼び出し、受け取った結果を演算し、結果を返すというものである。eval()の引数には、現在のプロセス変数の一覧が渡されるものとする。

プロセス変数	値
A	15
B	20
C	46



4　フィルター

4.1　クラス図



4.2　Filter

4.2.1　概要

ワークフローAPI 上で引数に指定する絞込条件を表すオブジェクトである。本ワークフローシステム運用基盤では、名前によるフィルタリング、状態によるフィルタリング、プロセス変数によるフィルタリング、グループフィルタリングをサポートする。なお、グループフィルタリングは、それに含まれるフィルターの論理積(AND)のみをサポートする。なおフィルター対象を特定する文字列がどのように利用されるかは、フィルターを利用する API によって異なる。

本ワークフローシステム運用基盤では、以下のフィルターを提供する。

フィルター	説　　明
NameFilter	名称でフィルタリングを行う。対象となる名称は、以下のとおりである。 プロセス定義名 プロセスインスタンス名 アクティビティインスタンス名
StateFilter	状態でフィルタリングを行う。対象となる状態は、以下のとおりである。 プロセス定義状態 プロセスインスタンス状態

	アクティビティインスタンス状態 作業項目状態
AttributeFilter	プロセス変数でフィルタリングを行う。
GroupFilter	複数のフィルターを持ち、それらのフィルターの論理積(AND)でフィルタリングを行う。

4.2.2 インタフェース

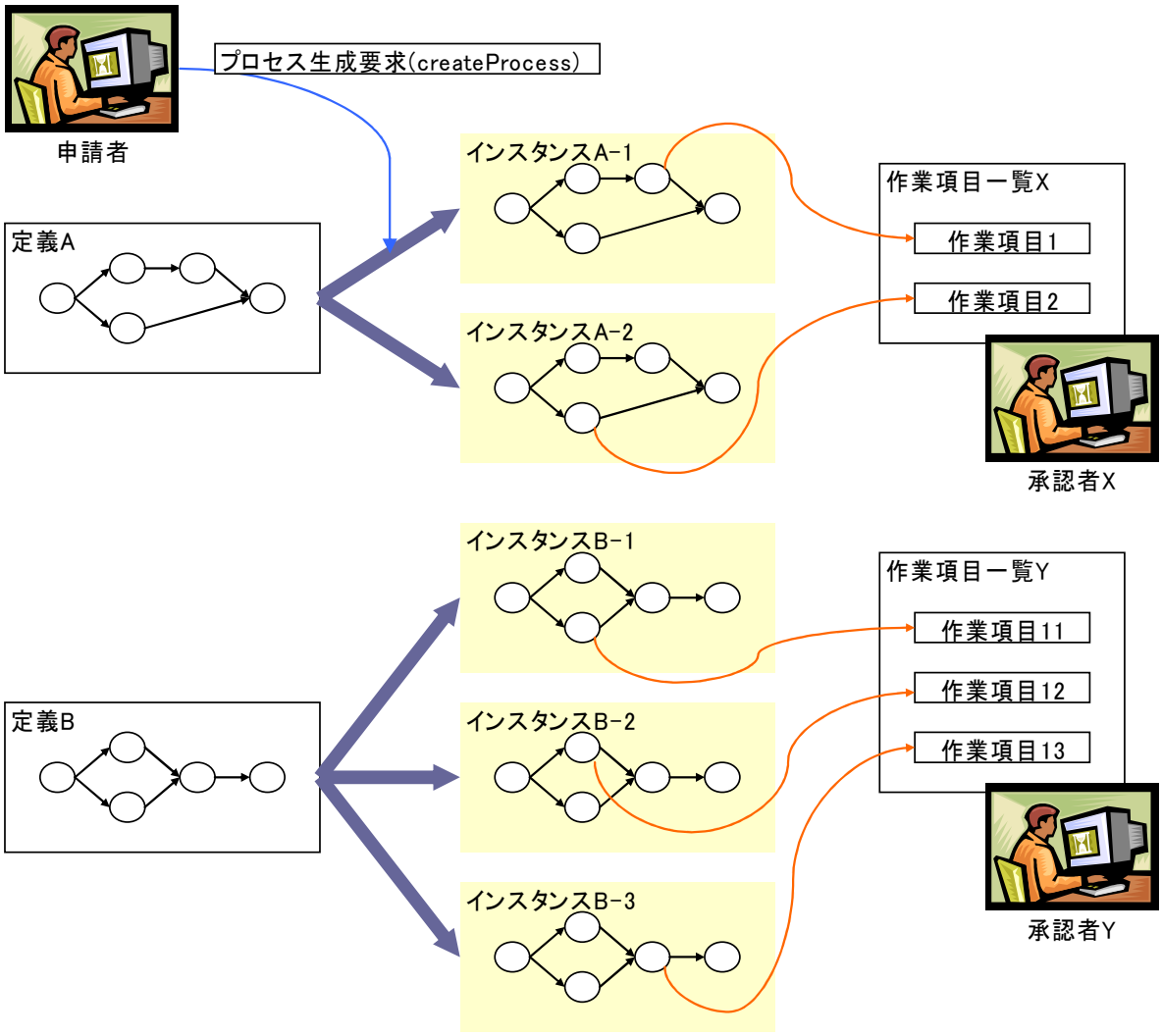
インタフェース	説　　明
int getType()	フィルターのタイプを表す。本ワークフローシステム運用基盤では、名前によるフィルタリング、状態によるフィルタリング、プロセス変数によるフィルタリング、グループフィルタリングをサポートする。
String getFilterName()	フィルター対象を特定する文字列を表す。
int getComparison()	フィルタリングの演算子を表す。==、!=に対応する数値定数。
int getFilterString()	フィルタリング文字列。

5　定義リポジトリ

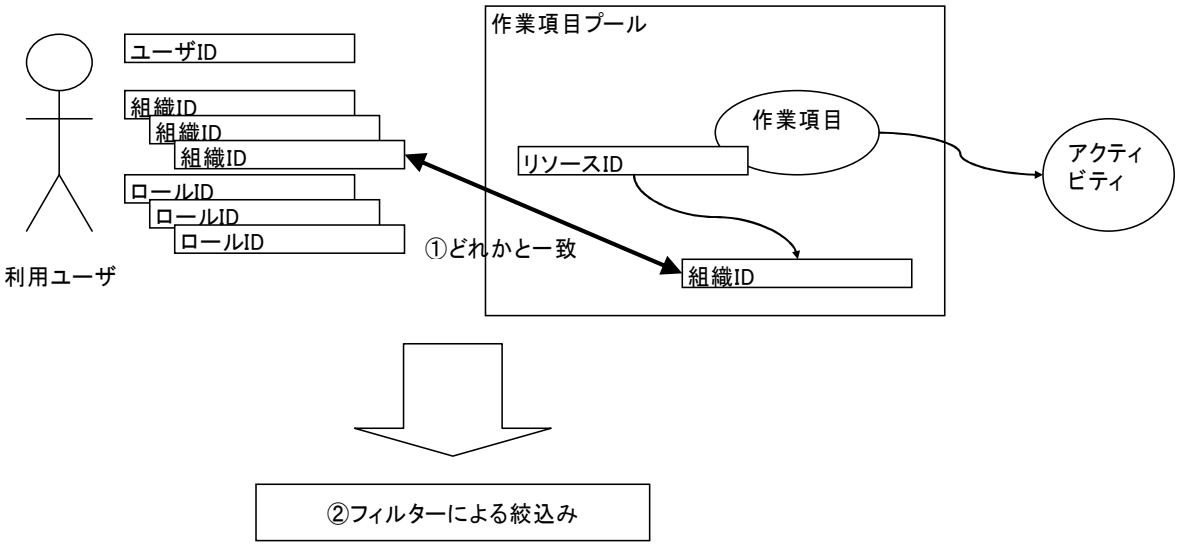
5.1　ワークフロー定義とインスタンスの関係

定義リポジトリは、ワークフロー定義を管理する。ワークフローエンジンは、そのワークフロー定義からインスタンスを生成してその状態を管理する。インスタンスの動作には定義の情報が必要であるため、いったん定義リポジトリに登録されたワークフロー定義は、基本的に変更と削除を許さないものとする。したがって、定義登録ツールにおいては既存の定義情報を更新しないようにするとともに、メンテナンス時において、不要になった定義を削除する場合は、同時に全インスタンスの情報も削除されなければならない。

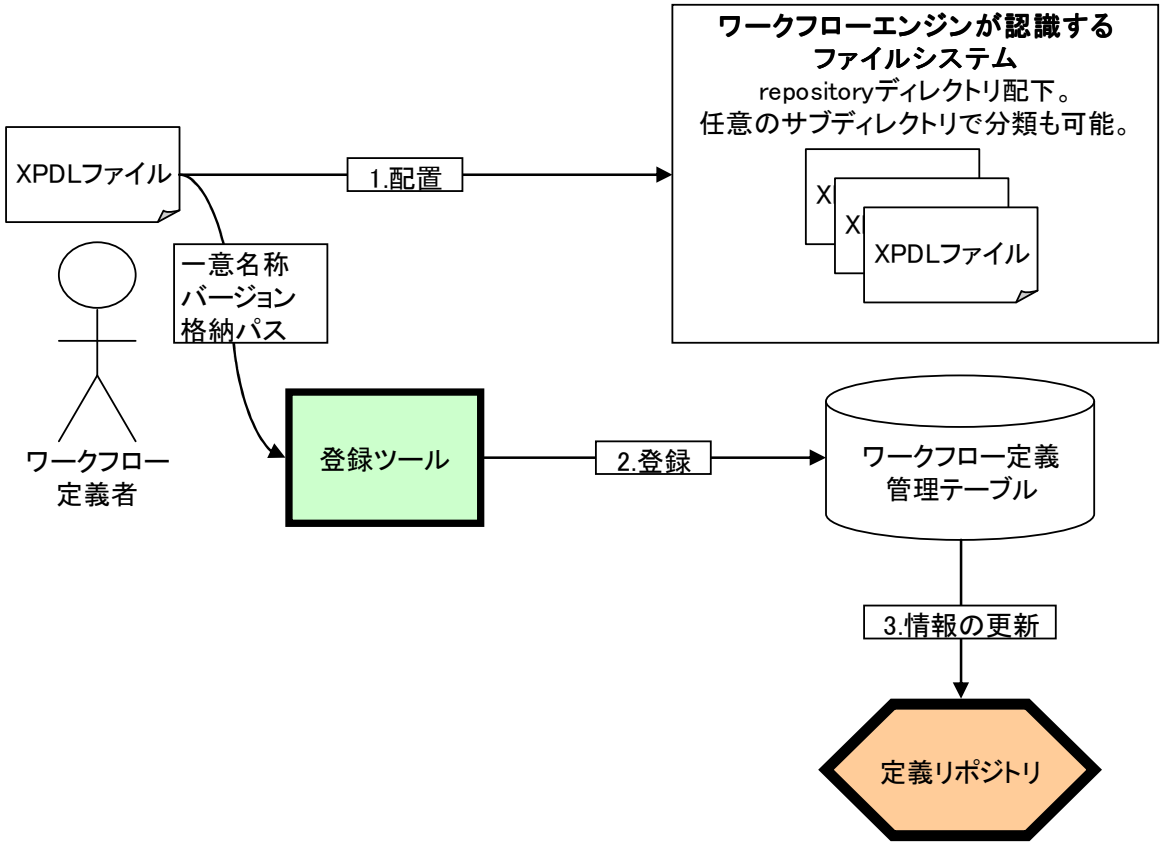
以下に、ワークフロー定義とインスタンス間の関係を、概念図として示す。ワークフロー定義を利用するクライアントアプリケーションがプロセス生成要求を行うと、定義に対応するインスタンスが生成される。インスタンスは、それぞれ独立して進行し、状態の遷移を行う。クライアントアプリケーションが進行に介在する場合は、検索条件によって絞り込まれた作業項目一覧から取得した作業項目に対する操作によって行われる。なお、作業項目とは、ワークフロー参加者とアクティビティを結びつける情報であり、プロセスインスタンス内の１つのアクティビティにおいて、ワークフロー参加者によって処理されるべき仕事を表現したものである。



5.1.1　作業項目の収集手順

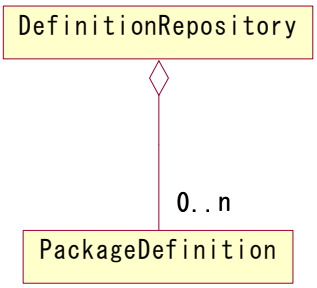


5.2　ワークフロー定義ファイルの配置



XPDL ファイルと名前のマッピングは、RDB に格納する。XPDL ファイルそのものは、プロパティ値 nautica.definitionRepository.root からの相対パスに格納される。その際、サブディレクトリを利用することで、定義ファイルを分類することが可能である。配置にあたっては、定義配置のためのツールを別途用意することとする。なお、アプリケーションサーバから使用する場合は、ポリシーファイルを編集し、適切な読み取りアクセス許可を設定するものとする。

5.3 クラス図



5.4 DefinitionRepository

5.4.1 概要

DefinitionRepository は、XPDL ファイルと、名称のマッピングを管理しており、

要求に応じて、XPDL をオブジェクト化して返す。

5.4.2 インタフェース

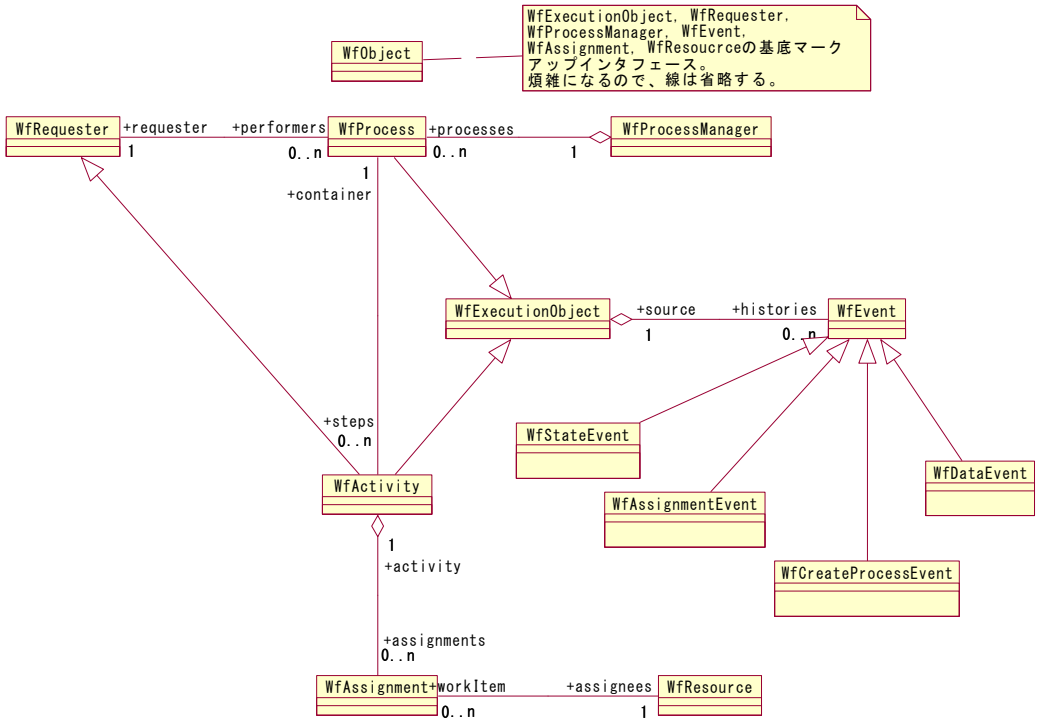
DefinitionRepository は以下のインタフェースを持つ。

インタフェース	説　　明
ProcessDefinition[] getProcessDefinitions(Session session, Filter filter)	プロセス定義一覧の取得
ProcessDefinitionState[] getProcessDefinitionStates (Session session, String pdid)	プロセス定義状態一覧の取得
void changeProcessDefinitionState (Session session, String pdid, ProcessDefinitionState state)	プロセス定義状態の変更

6　ワークフローエンジンのコアインタフェース

6.1　クラス図

ワークフローエンジンのコア部分には、OMG(Object Management Group)が公開しているワークフロー管理基盤(Workflow Management Facility)を参考にした以下のモデルを適用する。なお、ワークフロー管理基盤関連のドキュメントは、OMG および WfMC のサイトから入手可能である。



6.2　WfObject

6.2.1　概要

WfObject はコアインタフェースを構成する各要素の基底を表すマークアップ目的のインタフェースである。OMG の仕様では、BaseBusinessObject に相当するインタフェースである。

6.3　WfExecutionObject

6.3.1　概要

WfExecutionObject は、WfProcess と WfActivity の共通の属性や状態、操作を定義する管理実体の基底インタフェースである。本書では、この管理実体を「実行オブ

ジェクト」と表記する。

WfExecutionObject は、内部状態の取得および設定機能を提供する。現時点での状態を取得する操作、別の状態に遷移させる操作、ある特定の状態遷移に対する操作が提供されている。これらの操作は保留(suspend)、再開(resume)、停止(terminate)、中断(abort)である。インタフェースには、名前、説明、キーとなる属性を含む。また、実行履歴を表す監査情報を、指定されたフィルタに基づいて返すことによって、WfExecutionObject の実行の監視操作も提供する。他にも、コンテキストの取得と設定のメソッドがある。

6.3.2 インタフェース

WfExecutionObject は以下のインタフェースを持つ。

インタフェース	説　　明
String getID()	ID を返す。
String getName()	名称を返す。
void setName(String name)	名称を設定する。
String getDescription()	詳細情報を返す。
void setDescription(String desc)	詳細情報を設定する
String getState()	現在の状態を返す。
String[] getTransitableStates()	現在の状態から遷移可能な状態の一覧を返す。
void changeState(String state)	指定された状態に遷移を試みる。
String getLastStateTime()	最後に状態が遷移した時間を返す。
void resume()	保留中の実行オブジェクトの再開を要求する。要求が受け入れられた場合、状態は、open.not_running.suspended から、open.running に設定される。
void suspend()	実行オブジェクトの保留を要求する。要求が受け入れられた場合、状態は、open.not_running.suspended に設定される。
void terminate()	実行中の実行オブジェクトの停止を要求する。要求が受け入れられ

	た場合、状態は closed.terminated に設定される。
void abort()	保留された実行オブジェクトの中断を要求する。要求が受け入れられた場合、状態は closed.aborted に設定される。
ContextEntry[] getContext()	実行オブジェクトにかかわる属性一覧を返す。
void setContext(ContextEntry[] context)	実行オブジェクトにかかわる属性一覧を設定する。
int getHistoryCount()	変更履歴の数を返す。
WfEvent getHistory(int index)	インデクスを指定して、履歴を取得する。
WfEvent[] getAllHistories()	この実行オブジェクトに関わる変更履歴を取得する。

6.4 WfProcess

6.4.1 概要

WfProcess は、ひとつのビジネスプロセスの管理実体を表すインタフェースである。本書では、この管理実体を「プロセス」と表記する。WfProcess は、プロセスの実行を開始する操作、プロセスによって生成された結果を取得する操作、WfRequester や WfActivity との関連を追加することで、WfExecutionObject を特化している。

6.4.2 インタフェース

WfProcess は以下のインタフェースを持つ。

インタフェース	説　　明
WfRequester getRequester()	プロセス要求者を返す。
void setRequester(WfRequester requester)	プロセス要求者をプロセスに関連付ける。プロセスが生成された時点で、この関連付けが行われる。プロセスは、WfRequester のインタフェースを通じて、状態の変更を通知する。
WfProcessManager getManager()	プロセスは、WfProcessMgr に関連付けられる。WfProcess が生成される時点で関連付けが行われ、

	変更はできない。
int getStepCount()	0 以上の WfActiviy がプロセスに関連付けられる。ステップは、現時点でプロセス内で open な状態であるアクティビティを表すものとする。ここでは、ステップの個数を返す。
WfActivity getStep(int index)	ステップからインデクスで指定されたアクティビティを返す。
WfActivity[] getAllSteps()	現時点の全ステップを返す。
boolean isMemberOfStep(WfActivity act)	引数で指定されたアクティビティが、そのプロセスのステップに所属しているかどうかを返す。
WfActivity[] getActivitiesInState (String state)	引数で指定された状態を持つアクティビティを返す。
void start()	WfProcess の実行を開始するために使用される。プロセスの状態は、open.not_running から open.running に変化する。
ContextEntry[] getResult()	WfProcess によって生成される結果を返す。プロセスが完了するまでは結果は定義されない。結果の取得がまだ取得できない場合は、例外が発生する。

6.5 WfActivity

6.5.1 概要

WfActivityはビジネスプロセスにおける1つの作業要求ステップの管理実体を表すインタフェースである。本書では、この管理実体を「アクティビティ」と表記する。WfActivity は、1つの WfProcess に関連付けられている。任意の時点において、WfProcess 内では複数の WfActivity オブジェクトが活性化されている場合がある。

WfActivity は、アクティビティに対する明示的な完了操作と作業結果を設定する操作、WfProcess や WfAssignment との関連を追加することで、WfExecutionObject を特化している。

6.5.2 インタフェース

WfActivity は以下のインタフェースを持つ。

インタフェース	説　　明
WfProcess getContainer()	このアクティビティが所属している WfProcess を返す。
int getAssignmentCount()	割り当てられているアサインメントの個数を返す。
WfAssignment getAssignment(int index)	インデクスで指定されたアサインメントを返す。
WfAssignment[] getAllAssignment()	割り当てられているすべてのアサインメントを返す。
boolean isMemberOfAssignment(WfAssignment ass)	指定されたアサインメントが、このアクティビティに割り当てられたものかどうかを返す。
void complete()	WfActivity の完了を要求するために、アプリケーションによって使用される。これは、setResult()と共に、アクティビティの結果をプロセスに返すために使用される。
ContextEntry[] getResult()	WfActivity によって表される作業の実行によって生成される結果を返す。結果データがまだ利用可能でない場合は、例外が発生する。
void setResult(ContextEntry[] result)	WfActivity によって表される作業の実行によって生成される結果を設定する。

6.6 WfEvent

6.6.1 概要

WfEvent は、ワークフロー遂行中に発生するイベントの管理実体を表すインタフェースである。OMG 仕様では WfEventAudit としているが、WfEvent とした。本書では、この管理実体を「ワークフロー監査情報」と表記する。WfEvent は、すべての種類のワークフロー監査情報の基底インタフェースを提供する。イベントの種類には、状態の変化、リソース割り当ての変更、データの変更などがある。

また、ワークフロー監査情報はワークフローの変更履歴としての側面を持ち、永続

化されている必要がある。この変更履歴は、WfExecutionObject の getHistory()操作によってアクセス可能とする。WfEvent オブジェクトは、ワークフローがその状態を変化させる時に生成されるが、その存続期間はイベントを引き起こしたオブジェクトの存続期間とは独立して管理される。それらは、別途運用管理にて定めるものとする。

6.6.2 インタフェース

WfEvent は以下のインタフェースを持つ。

インタフェース	説　　明
String getEventType()	イベントの種類を返す。
WfExecutionObject getSource()	イベントを引き起こした WfExecutionObject を返す。
String getTtimestamp()	イベントが発生した日時を返す。
String getProcessManagerName()	イベントに関連するプロセスマネージャの名称を返す。
String getProcessManagerVersion()	イベントに関連するプロセスマネージャのバージョンを返す。
String getProcessID()	イベントに関連するプロセス ID を返す。
String getProcessName()	イベントに関連するプロセス名称を返す。
String getActivityID()	イベントに関連するアクティビティ ID を返す。
String getActivityName()	イベントに関連するアクティビティ名称を返す。

6.7 WfRequester

6.7.1 概要

WfRequester は、プロセスの実行や結果取得を行う管理実体を表す基底インタフェースである。本書では、この管理実体を「プロセス要求者」と表記する。プロセス要求者によって要求されたプロセスは、重要な状態変化をプロセス要求者に伝えるものとする。1 つのプロセス要求者は、複数のプロセスを要求し、管理することも想定しなければならない。

WfRequester を実装するクラスとして考えられるのは、ひとつはアクティビティである。サブプロセスを要求するアクティビティは、このインタフェースを実装する必要がある。もうひとつは、クライアントからのプロセス生成を可能にするための、アダプターが考えられる。

6.7.2 インタフェース

WfRequester は以下のインタフェースを持つ。

インタフェース	説　　明
int getPerformerCount()	管理しているプロセスの個数を返す。
WfProcess getPerformer(int index)	インデクスで指定されたプロセスを返す。
WfProcess[] getAllPerformers()	管理下のすべてのプロセスを返す。
boolean isMemberOfPerformer(WfProcess proc)	引数で指定されたプロセスが、管理下にあるかどうかを返す。
void receiveEvent(WfEvent event)	WfProcess がそのリクエスタにワークフローイベントを通知するために使用されるリスナーインタフェース。このインタフェースを通して、WfProcess は、完了(complete)、停止(terminate)、中断(abort)のイベントや、クローズド(closed)状態への遷移をリクエスタに通知するものとする。

6.8 WfResource

6.8.1 概要

WfResource は、アクティビティに割り当てられる人や物を表す管理実体を表す基底インタフェースである。本書では、この管理実体を「リソース」と表記する。WfResource は、WfAssignment によって WfActivity と関連付けられる。WfResource を実装するクラスとして考えられるのは、人や物を表すオブジェクトに対するアダプタなどである。

6.8.2 インタフェース

WfResource は以下のインタフェースを持つ。

インタフェース	説　　明
int getWorkItemCount()	リソースに関連付けられたアサインメントの個数を返す。
WfAssignment getWorkItem(int index)	インデクスで指定されたアサインメントを返す。

WfAssignment[] getAllWorkItem()	リソースに関連付けられたすべてのアサインメントを返す。
boolean isMemberOfWorkItems(WfAssignment ass)	引数で指定されたアサインメントと、関連を持つかどうかを返す。
String getID()	ID を返す。
String getName()	名称を返す
void release(WfAssignment ass, String info)	引数に指定したアサインメントにとって、リソースが不要になったことを通知するリスナーインタフェース。引数として、もうリソースに関連付けられていないアサインメントと、リソースを解放する理由をとる。 このインタフェースは、アサインメントが削除された場合や、アサインメントが他のリソースへ再割り当てされた場合に内部的に呼び出されることを想定している。

6.9 WfAssignment

6.9.1 概要

WfAssignment は、WfActivity オブジェクトを WfResource オブジェクトに結合させる管理実体を表すインタフェースである。本書では、この管理実体を「アサインメント」と表記する。アサインメントは、リソース管理機能によって管理されることを想定しており、アクティビティとリソースを使用して適切にアサインメントを生成・管理する。

WfAssignment は、アクティビティの実行準備を整える段階で、リソース選択過程の一部として生成される。WfAssignment の存続期間は、関連付けられたアクティビティの存続期間によって制限される。

6.9.2 インタフェース

WfAssignment は以下のインタフェースを持つ。

インタフェース	説　　明
WfActivity getActivity()	リソースを割り当てるアクティビティを返す。
WfResource getAssignee()	割り当てられたリソースを返す。

void setAssignee(WfResource new_value)	指定されたリソースを設定する。

6.10 WfProcessManager

6.10.1 概要

WfProcessManager は、特定のワークフロープロセスのテンプレートを表すインタフェースである。OMG 仕様では WfProcessMgr としているが、WfProcessManager とした。本書では、この管理実体を「プロセスマネージャ」と表記する。WfProcessManager は、WfProcess インスタンスのファクトリである。プロセスマネージャは、プロセスが必要とする情報や、プロセス定義情報などに対するアクセスを提供する。プロセスマネージャは、プロセス定義のラッパーオブジェクトとして機能する。

6.10.2 インタフェース

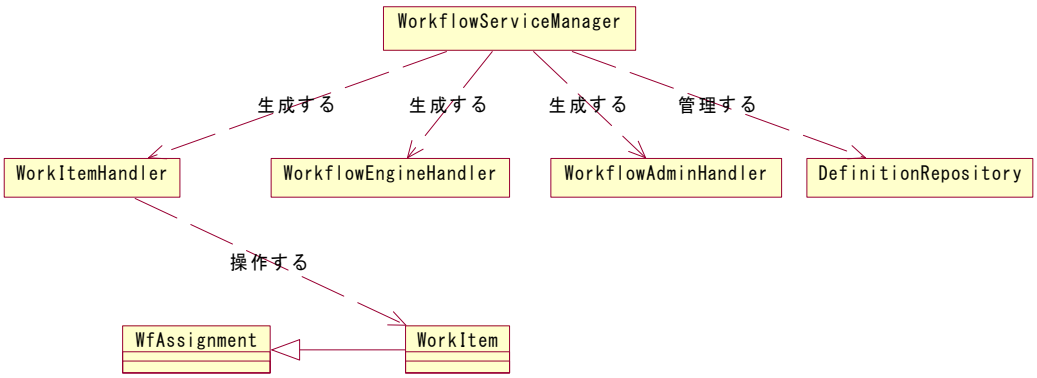
WfProcessManager は以下のインタフェースを持つ。

インタフェース	説　　明
int getProcessCount()	管理しているプロセスの個数を返す。
WfProcess getProcess(int index)	インデクスで指定されたプロセスを返す。
WfProcess[] getAllProcesses()	管理下のすべてのプロセスを返す。
boolean isMemberOfProcess(WfProcess member)	引数で指定されたプロセスが、管理下にあるかどうかを返す。
String getCategory()	カテゴリーを返す。
String getName()	名称を返す。
String getVersion()	バージョンを返す。
String getDescription()	詳細情報を返す。
String getState()	状態を返す。新規のプロセス生成の可否を返す。
void setState(String state)	状態を設定する。
ContextEntry[] getContextSignature()	インスタンスのコンテキストの設定方法を定義するメタ情報を返す。
ContextEntry[] getResultSignature()	インスタンスが結果を返す方法を指定するメタ情報を返す。
WfProcess createProcess(WfRequester requester)	プロセスのインスタンスを生成

	し、リクエスタに結合する。プロセスが生成されると、not_running.not_started の状態に設定される。

7　ワークフローサービス管理

7.1　クラス図



7.2　WorkflowServiceManager

7.2.1　概要

ワークフローサービスマネージャは、ワークフローに対するクライアントインタフェースである。

7.2.2　インタフェース

インタフェース	説　　明
Session connect(ConnectionInfo ci)	ワークフローエンジンへの接続
void disconnect(Session session)	ワークフローエンジンとの切断
DefinitionRepository getDefinitionRepository(Session session)	定義リポジトリを返す。
WorkItemHandler getWorkItemHandler(Session session)	作業項目ハンドラを返す。
WorkflowEngineHandler getWorkflowEngineHandler(Session session)	ワークフローエンジンハンドラを返す。
WorkflowAdminHandler getWorkflowAdminHandler(Session session)	ワークフロー管理ハンドラを返す。

7.3　WorkflowEngineHandler

7.3.1　概要

ワークフローエンジンハンドラは、ワークフローに対するクライアントインタフェースである。

7.3.2 インタフェース

インタフェース	説　　明
String createProcess (Session session, String pdid, String name)	プロセスインスタンスの生成
String startProcess (Session session, String pid)	プロセスインスタンスの開始
void terminateProcess (Session session, String pid)	プロセスインスタンスの停止
ProcessState[] getProcessStates(Session session, String pid, Filter filter)	プロセスインスタンス状態一覧の取得
void changeProcessState(Session session, String pid, ProcessState state)	プロセスインスタンス状態の変更
Attribute[] getProcessAttributes(Session session, ProcessID pid, Filter filter)	プロセスインスタンス属性一覧の取得
Attribute getProcessAttributeValue(Session session, String pid, String name)	プロセスインスタンス属性の属性値の取得
void assignProcessAttribute(Session session, String pid, Attribute attr)	プロセスインスタンス属性の割り当て
ActivityState[] getActivityStates(Session session, String pid, String aid,	アクティビティインスタンス状態一覧の取得

Filter filter)	
void changeActivityInstanceState(Session session, String pid, String aid, ActivityState state)	アクティビティインスタンス状態の変更
Attribute[] getActivityAttributes(Session session, String pid, String aid, Filter filter)	アクティビティインスタンス属性一覧の取得
Attribute getActivityAttributeValue(Session session, String pid, String aid, String name)	アクティビティインスタンス属性の属性値の取得
void assignActivityAttribute(Session session, String pid, String aid, Attribute attr)	アクティビティインスタンス属性の割り当て
Process[] getProcesses(Session session, Filter filter)	プロセスインスタンス一覧の取得
Process getProcess(Session session, String pid)	プロセスインスタンスの取得
Activity[] getActivities(Session session, Filter filter)	アクティビティインスタンス一覧の取得
Activity getActivity(Session session, String pid, String aid)	アクティビティインスタンスの取得

7.4 WorkItemHandler

7.4.1 概要

作業項目ハンドラは、ワークフローに対するクライアントインタフェースである。

7.4.2 インタフェース

インタフェース	説　　明
WorkItem[] getWorkItems(Session session, Filter filter)	作業項目一覧の取得
WorkItem getWorkItem(Session session, String wid)	作業項目の取得
void completeWorkItem(Session session, String wid)	作業項目の完了
WorkItemState[] getWorkItemStates(Session session, String pdid, Filter filter)	作業項目状態一覧の取得。現在の作業項目が遷移可能な状態の一覧を返す。状態がない場合は、空の配列を返す。
WorkItemState[] changeWorkItemState (Session session, String wid, ProcessDefinitionState state)	作業項目状態の変更
void reassignWorkItem (Session session, Participant sourceUser, Participant targetUser, String wid)	作業項目の再割り当て
Attribute[] getWorkItemAttributes(Session session, String wid, Filter filter)	作業項目属性一覧の取得
Attribute getWorkItemAttribute(Session session,	作業項目属性の取得

String wid, String name)	
void assignWorkItemAttribute(Session session, String wid, Attribute attr)	作業項目属性の割り当て

7.5 WorkflowAdminHandler

7.5.1 概要

ワークフロー管理ハンドラは、ワークフローに対する管理クライアントインタフェースである。

7.5.2 インタフェース

インタフェース	説　　明
void changeProcessesState(Session session, String pdid, Filter filter, ProcessState state)	プロセスインスタンス状態の変更
void changeActivitiesState(Session session, String pdid, String adid, Filter filter, ActivityState state)	アクティビティインスタンス状態の変更
void terminateProcesses(Session session, String pdid, Filter filter)	プロセスインスタンスの停止
void assignProcessesAttribute(Session session, String pdid, Filter filter, Attribute attr)	プロセスインスタンス属性の割り当て
void assignActivitiesAttribute(	アクティビティインスタンス属性

Session session, String pdid, String adid, Filter filter, Attribute attr)	の割り当て
void abortProcesses(Session session, String pdid, Filter filter)	複数のプロセスインスタンスの中断
void abortProcess(Session session, String pid)	プロセスインスタンスの中断

7.6 WorkItem

7.6.1 概要

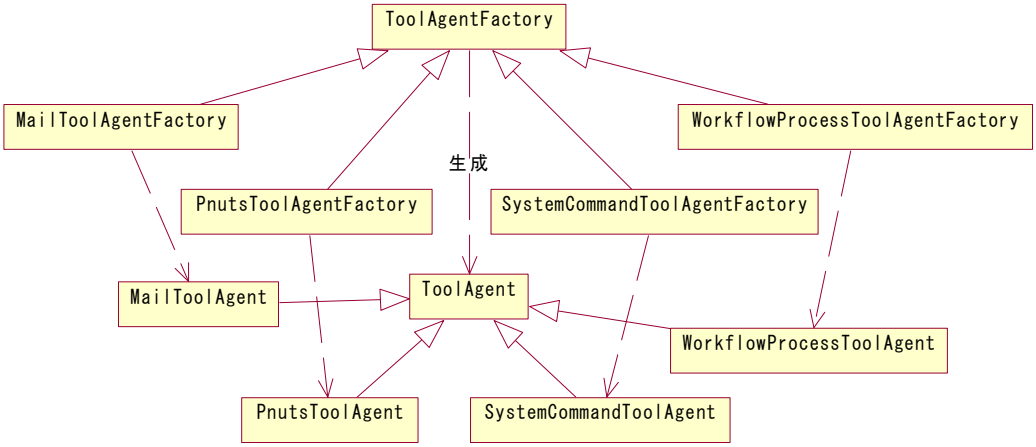
作業項目をあらわす。作業項目とは、ワークフロー参加者とアクティビティを結びつける情報であり、プロセスインスタンス内の１つのアクティビティにおいて、ワークフロー参加者によって処理されるべき仕事を表現したものである。ユーザは、作業項目の ID を通じて、プロセス変数の取得、更新を行い、作業を完了させるものとする。

7.6.2 インタフェース

インタフェース	説　　明
String getID()	ID を返す。

8　ツールエージェント

8.1　クラス図



8.2　ToolAgentFactory

8.2.1　概要

ToolAgentFactory は、各種 ToolAgent を生成するためのファクトリの基底インタフェースである。

8.2.2　インタフェース

インタフェース	説　　明
ToolAgent getToolAgent()	初期化されたツールエージェントを返す。

8.3　ToolAgent

8.3.1　概要

ToolAgent は、ワークフローアプリケーションの実行を管理するエージェントの基底インタフェースである。これは、WfMC のインタフェース 3 に示されるツールエージェントの概念に該当する。個別のツールエージェントの実装方法はここでは規定しないが、WfMC の仕様書からは、内部に実行要求のための待ち行列のようなものを想定しているようである。本ワークフローシステム運用基盤で標準提供するツールエージェントの個別実装は、この考え方を適用する。

なお、本ワークフローシステム運用基盤では、以下に示すツールエージェントを標準提供する。

ツールエージェント	説　　明
SystemCommandToolAgent	システムコマンドを起動するようなツールエージェントである。
MailToolAgent	メール送信を行うツールエージェントである。
PnutsToolAgent	Pnuts スクリプトを実行するツールエージェントである。
WorkflowProcessToolAgent	指定されたワークフローエンジンにプロセス生成を行うツールエージェントである。エンジン間接続も参照すること。

8.3.2　インタフェース

インタフェース	説　　明
int invokeApplication(String appName, String wid, Attribute[] attrList, int mode)	アプリケーションの起動
int requestAppStatus(int toolAgentHandle, String appName, String wid, AttributeList attrList)	アプリケーション状態の要求
void terminateApplication(int toolAgentHandle, String wid)	アプリケーションの停止

8.3.3　設定ファイル

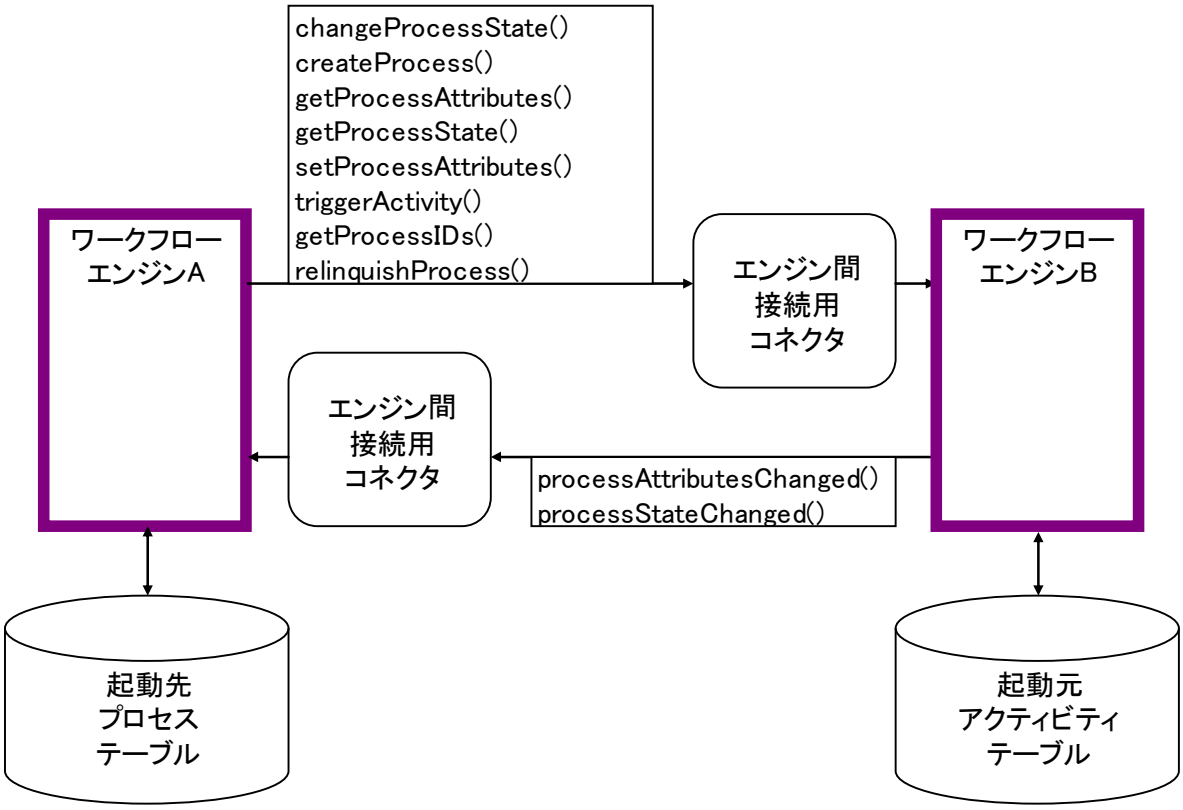
ツールエージェントとアプリケーションのマッピングテーブル。個別のツールエージェントの設定ファイルは、必要に応じて定義する。

```
<tool-agents>
<agent classname=”xxxxxxxx”>
<app name=”aaaa”/>
<app name=”bbbb”/>
<app name=”cccc”/>
</agent>
<agent classname=”yyyyyyy”>
<app name=”ddd”/>
<app name=”eee”/>
```

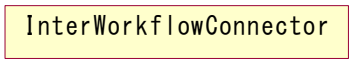
```
</agent>
</tool-agents>
```

9　エンジン間接続

9.1　概念図



9.2　クラス図



9.3　InterWorkflowConnector

9.3.1　概要

エンジン間接続のためのインタフェースとして、InterWorkflowConnector を提供する。

9.3.2　インタフェース

インタフェース	説　　明
String connect(String seid, String password)	自エンジンの ID を指定して、リモートエンジンアクセス用の ID を返す。

void disconnect(String eid)	リモートのエンジンとの接続を切断する。
void changeProcessState(String eid, String pid, ProcessState state)	プロセスインスタンス状態の変更
String createProcess(String eid, String pdid, int flag, String parentid, String aid, String userid, String roleid)	プロセスインスタンスの生成
Attribute[] getProcessAttributes(String eid, String pid, String rootid, String aid)	プロセスインスタンス属性の取得
ProcessState getProcessState(String eid, String pid)	プロセスインスタンス状態の取得
void processAttributesChanged(String eid, String pid, Attribute[] attr)	プロセスインスタンス属性の変更通知
void processStateChanged(String eid, String pid, ProcessState state)	プロセスインスタンス状態の変更通知
void setProcessAttributes(String eid, String rootid, String aid, String pid, Attrbute[] attr)	プロセスインスタンス属性の設定

void triggerActivity(String eid, String rootid, String aid, String pid)	アクティビティに対するトリガー
String[] listProcesses(String eid, Filter filter)	プロセスインスタンス一覧の取得
void relinquishProcess(String eid, String pid)	プロセスインスタンスの放棄

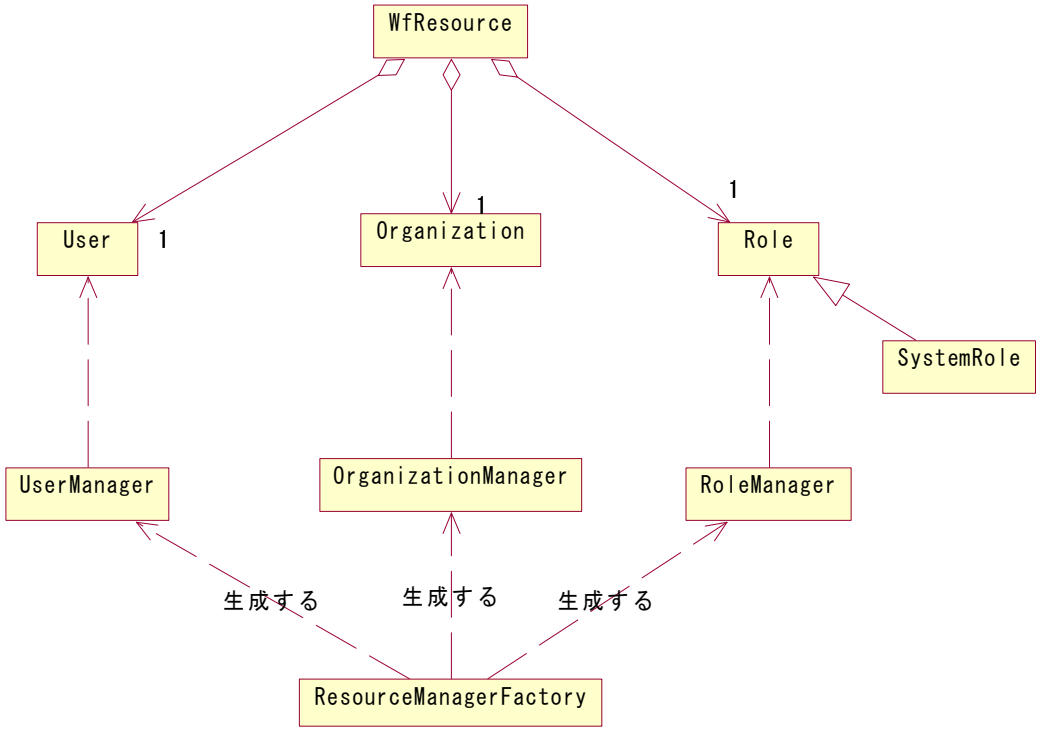
9.3.3 設定ファイル

```
<engine-config>
<engine id="xxx" password="passxxx" />
<remote-engines>
<engine id="aaa" password="password" url="iiop://aaa.jp:1050" />
<engine id="bbb" password="password" url="iiop://bbb.jp:1050" />
</remote-engines>
</engine-config>
```

10　ワークフロー参加者管理

10.1 クラス図

本ワークフローシステム運用基盤では、ワークフロー参加者管理に対して以下のようなインタフェースを提供する。



10.2 ResourceManagerFactory

10.2.1　概要

ResourceManagerFactory は、各種管理をとりだすためのファクトリークラスである。任意のリソース管理システムを利用するために、このクラスを利用して具体実装を取得しなければならない。対応するユーザ管理、組織管理、権限管理もそれぞれ個別実装を持つものとする。これらは順に nautica.userManager 、nautica.organizationManager、nautica.roleManager の各プロパティで指定されたクラスを使用する。本ワークフローシステム運用基盤では、参考実装を提供する。

10.2.2　インタフェース

インタフェース	説　　明
UserManager getUserManager()	ユーザ管理を返す。

OrganizationManager getOrganizationManager()	組織管理を返す。
RoleManager getRoleManager()	権限管理を返す。

10.3 UserManager

10.3.1 概要

UserManager は、ユーザ管理関連の各種 API を定義する。実際には、任意のユーザ管理システムを利用するために、このインタフェースを利用した具体実装を用意しなければならない。

10.3.2 インタフェース

インタフェース	説　　明
User createUser(String uid, String password, Attribute[] attr)	新規にユーザを生成する。ユーザ ID が既に存在する場合には例外を返す。生成されたユーザ情報は未認証状態である。
void updateUser(User user)	ユーザの情報を更新する。
void deleteUser(User user)	ユーザを削除する。
User getUser(String uid)	uid に対応するユーザを返す。取得されるユーザ情報は未認証状態である。
User[] getAllUsers()	全ユーザを返す。取得されるユーザ情報は未認証状態である。
boolean checkUserExistence(String uid)	ユーザが存在するかどうかを返す。

10.4 OrganizationManager

10.4.1 概要

OrganizationManager は、組織管理関連の各種 API を定義する。実際には、任意の組織管理システムを利用するために、このインタフェースを利用した具体実装を用意しなければならない。

10.4.2 インタフェース

インタフェース	説　　明
Organization createOrganization(Attribute[] attr)	新規に組織を登録する。組織 ID は自動採番されるものとする。
void updateOrganization(Organization org)	組織情報を更新する。
void deleteOrganization(Organization org)	組織を削除する。組織に下位組織が割り当てられている場合、それ

	らも削除される。ユーザが割り当てられている場合には、
Organization getOrganization(String oid)	組織を返す。
Organization[] getAllOrganizations()	全組織を返す。
boolean checkOrganizationExistence(String oid)	組織が存在するかどうかを返す。
Organization[] getAllOrganizations(String oid)	指定した組織配下の全組織を返す.配列には指定した組織も含む。
void addAlias(String key, Organization org)	指定した別名で、論理的頂点組織を登録する。キー名が既に使われている場合は例外とする。
Organization getOrganizationByArias(String alias)	指定した別名に対応する組織を返す。キー名に対応する組織がない場合は例外とする。
void assignOrganization(Organization assigner, Organization assignee)	組織に指定された組織を割り当てる。
void resignOrganization(Organization resignee)	指定された組織を解除する。
Organization getHigherOrganization(Organization org)	上位組織を返す。
Organization[] getLowerOrganizations(Organization org)	下位組織のコレクションを返す。
void assignUser(Organization org, User user)	組織に指定されたユーザを割り当てる。
void ressignUser(Organization org, User user)	組織から指定されたユーザを解除する。
User[] getAssignedUser(Organization org)	組織に属しているユーザのコレクションを返す。
Organization[] getOrganizationByUser(User user)	ユーザの所属する組織を返す。

10.5 RoleManager

10.5.1 概要

RoleManager は、権限管理関連の各種 API を定義する。実際には、任意の権限管理システムを利用するために、このインタフェースを利用した具体実装を用意しなければならない。

10.5.2 インタフェース

インタフェース	説　　明
void registerRole(String role)	ロールを登録する。

void eraseRole(String role)	ロールを削除する。
void attachRole(Organization org, String role)	組織にロールを付与する。
void attachRole(User user, String role)	ユーザにロールを付与する。
void detachRole(Organization org, String role)	組織からロールを剥奪する。
void detachRole(User user, String role)	ユーザからロールを剥奪する。
String[] getAllRoles()	全ロールを返す。
String[] getAttachedRoles(Organization org)	組織に付与されている全ロールを返す。
String[] getAttachedRoles(User user)	ユーザに付与されている全ロールを返す。

10.6 User

10.6.1 概要

User は、ユーザを表す。

10.6.2 インタフェース

インタフェース	説　　明
String getID()	一意なユーザ ID を返す。
String getAttribute(String name)	引数で指定した情報名に対応するユーザ情報を返す。
void updateAttribute(String name, String attr)	引数で指定した情報名に対応するユーザ情報を更新する。あらかじめデータベース上に用意された以外の情報名を使用すると、例外となる。
String[] getAllAttributeNames()	読み取り可能なすべての情報名を返す。
void authenticate(String password)	引数に指定されたハッシュテーブルに格納された情報をもとに認証を行う。認証された場合は、内部状態が認証済みに変更される。
void changeAuthenticationInfo(String oldPassword, String newPassword)	ユーザの認証用情報を変更する。

10.7 Organization

10.7.1 概要

Organization は、組織を表すクラスである。

10.7.2 提供インタフェース

インタフェース	説　　明
String getID()	組織 ID を返す。
String getAttribute(String name)	組織が保持する属性を返す。
String[] getAllAttributeNames()	組織が保持するすべての属性名を返す。
void updateAttribute(String name, String attr)	組織の属性を更新する。

10.8 Role

10.8.1 概要

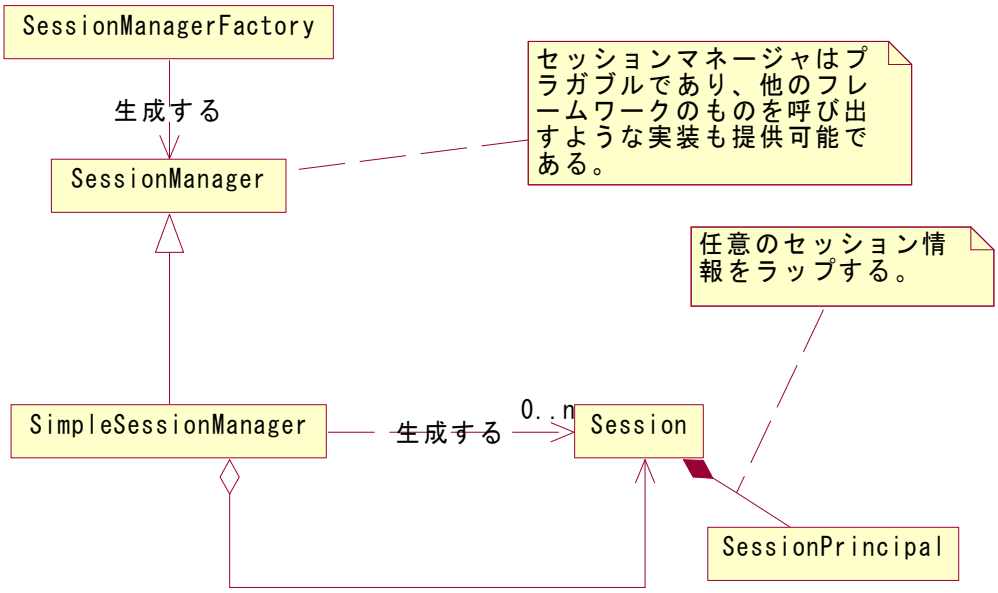
Role は、権限を表すクラスである。

10.8.2 提供インタフェース

インタフェース	説　　明
String getID()	権限 ID を返す。
String getName()	名称を返す。

11　セッション管理

11.1 クラス図



11.2 SessionManagerFactory

11.2.1　概要

セッション管理を生成するファクトリークラスである。任意のセッション管理システムを利用するために、このクラスを利用してセッションマネージャの具体実装を取得しなければならない。エンジンで使用するセッションマネージャは、プロパティ nautica.sessionManager に指定されたクラスを使用する。

11.2.2　提供インタフェース

インタフェース	説　　明

SessionManager getSessionManager()	セッションマネージャを返す。

11.3 SessionManager

11.3.1 概要

セッション管理を行う。実際には、任意のセッション管理システムを利用するために、このインタフェースを利用した具体実装を用意しなければならない。本ワークフローシステム運用基盤では、参考実装として SimpleSessionManager を提供する。

11.3.2 提供インタフェース

インタフェース	説　　明
String validateSession(String sid)	引数として渡されたセッション ID を検証する。セッション ID が null の場合は、新たなセッションを作成する。セッション ID に対応するセッション情報があれば、最終アクセス時間を更新する。戻り値として、対応付けたセッション ID を返す。対応するセッション情報がなければ、例外を返す。ここで新規にロールが割り当てられることはない。

String validateSession(String sid, String uid, String password)	引数として渡されたセッション ID を検証する。セッション ID が null の場合は、新たなセッションを作成する。セッション ID に対応するセッション情報(SessionPrincipal)があれば、最終アクセス時間を更新する。その後、引数のユーザ ID とパスワードを用いて、認証とロールの取得を行う。ユーザ情報とロールの組をセッション情報として、セッション ID に対応付ける。戻り値として、対応付けたセッション ID を返す。
void invalidateSession(String sid)	引数に渡されたセッション ID と、それに対応するセッション情報を無効にする。

11.4 Session

11.4.1 概要

ワークフローシステム運用基盤におけるセッション情報。任意のセッション管理に対応するため、既存のセッション情報をラップする形で提供される。

11.4.2 インタフェース

インタフェース	説　　明
String getID()	セッション ID を返す。
String getValue(String key)	キーに対応する情報を文字列形式で返す。

11.5 SimpleSessionManager

11.5.1 概要

このクラスは、セッションを管理するサンプル実装である。セッション情報は、セッション ID によって関連づけられている。セッション ID はエンジン内で一意であることが保障されている。

11.6 SessionPrincipal

11.6.1 概要

SessionPrincipal は、SimpleSessionManager 用のセッション情報を表すクラスである。

11.6.2 提供インタフェース

インタフェース	説　　明
String getID()	このセッション情報の ID
User getUser()	このセッションに割り付けられたユーザ情報
Date getCreatedTime()	このセッション情報の生成時間
Date getLastAccessedTime()	このセッション情報への最後のアクセス時間
public String getAttribute(String key)	キーで指定された任意のオブジェクトを取得する。
public void setAttribute(String key, String value)	指定したキーで業務ドメインに任意の値を登録する。同じキーに既に属性が割り当てられている場合は上書きする。
public boolean containsAttributeKey(String key)	キーで指定された値があるかどうかを返す。
String getAllAttributeKeys()	すべてのキー名を返す。

12　ロギング

12.1 概要

ロギングは、Jakarta プロジェクトの log4J を使用する。

13　各機能説明

13.1 ワークフローエンジン

13.1.1　処理要求管理機能

(1) 概要

ワークフローエンジンへの処理要求受付機能。処理完了を待たずに制御を戻せるプロセス生成要求、アクティビティ完了通知等は、要求や通知をキューに登録した後、直ちに呼び出し側に制御を戻す。受け付けた処理要求は、ワークフロー状態遷移の起点として使用する。

(2) 利用ソフトウェア

本機能の実現にあたって、メッセージキューの使用が不可欠であることから、以下のオープンソース製品を検討した。JORAM がかなり整備されているが、Apache Geronimo プロジェクトに採用された将来性、他の Apache（および Jakarta）プロジェクト関連との親和性、提供機能などを総合的に判断し、本プロジェクトでは ActiveMQ を採用することとする。ドキュメントについてはまとまった形では提供されていないが、基本的に JMS 1.1 標準インタフェースの範囲内での開発を想定しているため、情報収集は運用面に限定されると考えている。また、メッセージの永続化については、ActiveMQ のドキュメントにてパフォーマンス面で推奨されていることから、付属の BerkeleyDB 1.5 を使用することとする。

製品名	JMS 1.1 対応	永続化対応	提供ドキュメント	採用
OpenJMS 0.7	×	△	△	
ActiveMQ 2.0	○	○	△	○
JORAM 4.1	○	○	○	

(3) メッセージキューへの登録

メッセージキューへの登録は、JMS 1.1 インタフェースを利用して行うものとする。登録されるメッセージは必ず永続化指定を行うこととし、障害復旧時の配信リカバリを行えるようにする。

キューは、使用目的にあわせて以下のものを用意する。

- プロセス開始待ち行列
- プロセス完了待ち行列
- アクティビティ開始待ち行列
- アクティビティ完了待ち行列

(4) メッセージの受信

メッセージ受信は、JMS 1.1 インタフェースを利用して行うものとする。その際、MessageListener インタフェースを採用し、onMessage()メソッドにより、メッセージを受信するものとする。

13.1.2 状態遷移管理機能

(1) 概要

ワークフローの現在状態を管理し、受け付けた処理要求の種類に応じて、ワークフロー定義のワークフロー参加者情報を参照して、正当性を確認した後、以下に示すプロセス及びアクティビティの遷移と管理を行う。その際、他のエンジンと連携する場合には、ワークフローエンジン連携用 API を利用する。

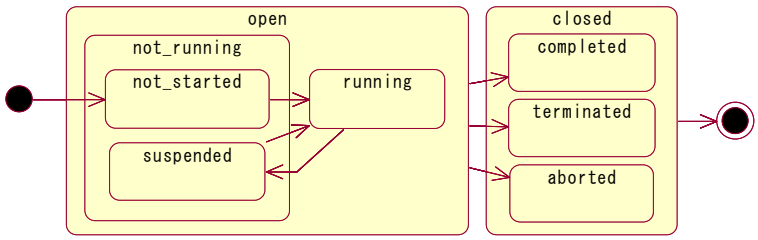
(2) 状態遷移

プロセス、アクティビティおよび作業項目は、ドットで区切られた階層構造の状態を持つものとする。

	状　　態		説　　明
1	open.not_running.not_started	開始待ち	インスタンスが生成されているが、実行が開始していない。遷移可能な状態は、3, 4, 5, 6。
2	open.not_running.suspended	保留中	実行が保留されていることを表す。この状態は、主に管理者によって人為的に引き起こされ、同時に付随するプロセス、アクティビティの保留も行う。遷移可能な状態は、3, 4, 5, 6。
3	open.running	実行中	実行中。遷移可能な状態は、2, 4, 5, 6。
4	closed.completed	完了	プロセス、アクティビティの正常終了を表す。遷移可能な状態はない。
5	closed.terminated	中断	プロセス、アクティビティの中断を表す。この状態は、主に管理者によって人為的に引き起こされ、同時に付随するプロセス、アクティビティの中断も行う。遷移可能な状態はない。

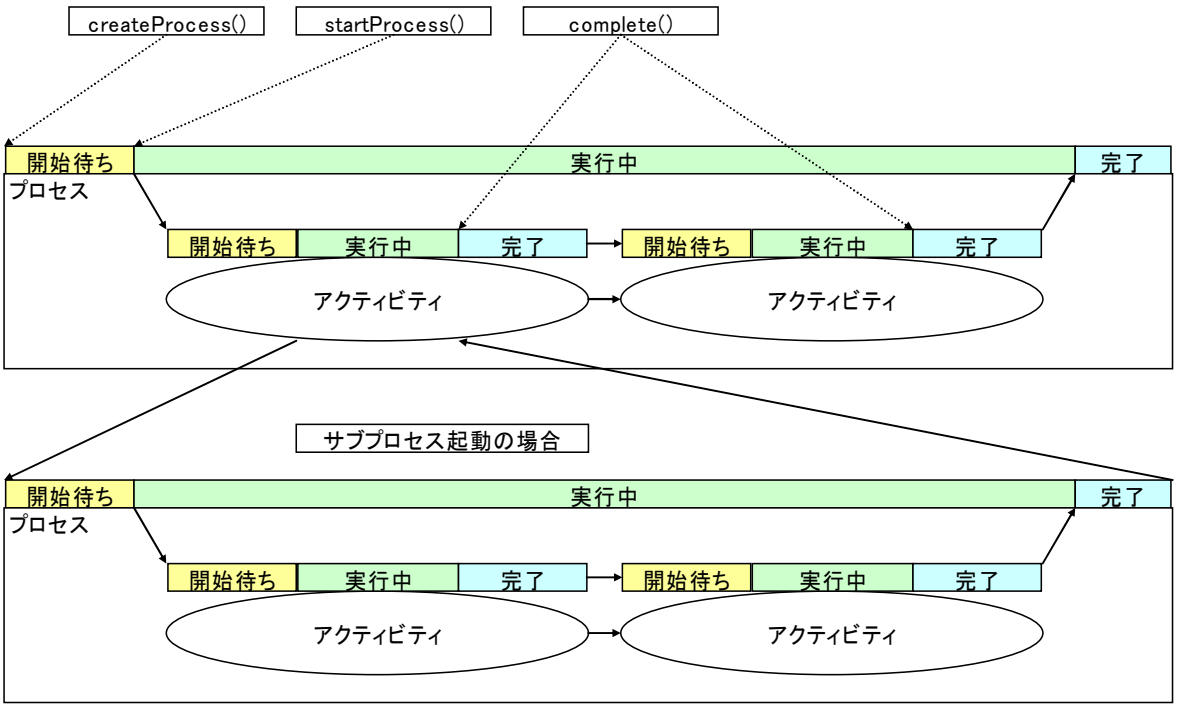
6	closed.aborted	中止	プロセス、アクティビティの中止。この状態は、例外発生などの異常事態を表し、付随するプロセス、アクティビティの状態変更は伴わない。異常事態であり、管理者による復旧作業を必要とする。遷移可能な状態はない。

これらの各状態間の遷移図を以下に示す。

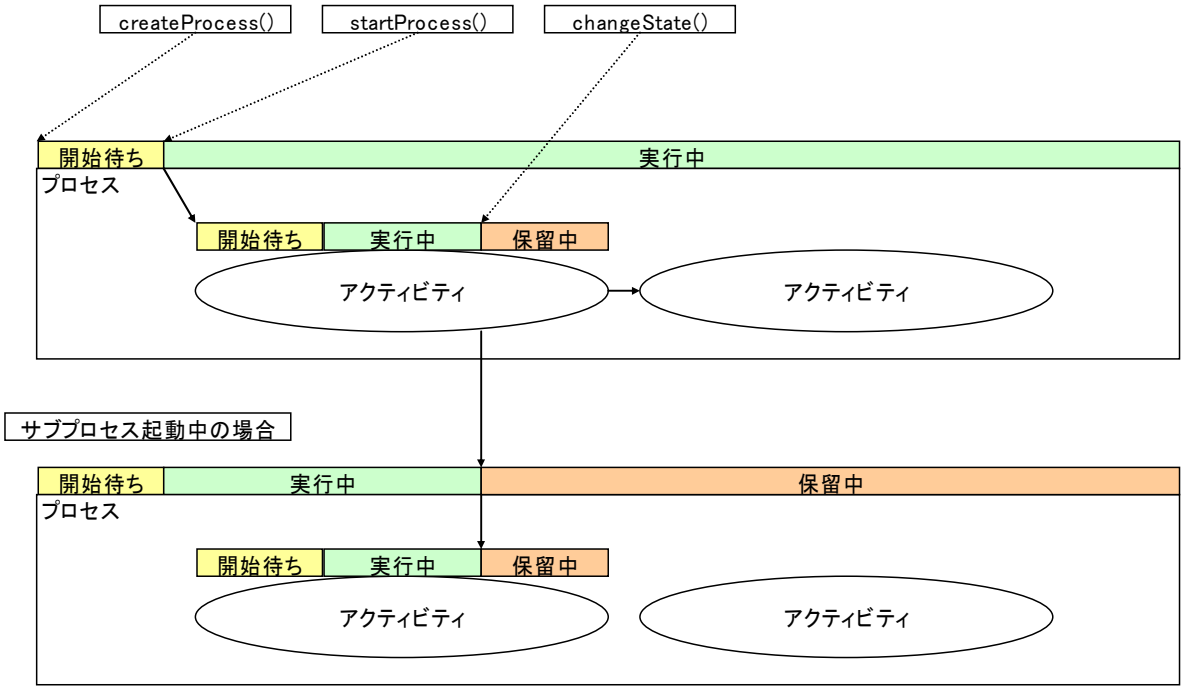


また、正常系の基本的な状態遷移は、open.not_running.not_started から始まり、実行時には open.running に移行し、完了後には closed.completed となり、状態遷移を完了する。

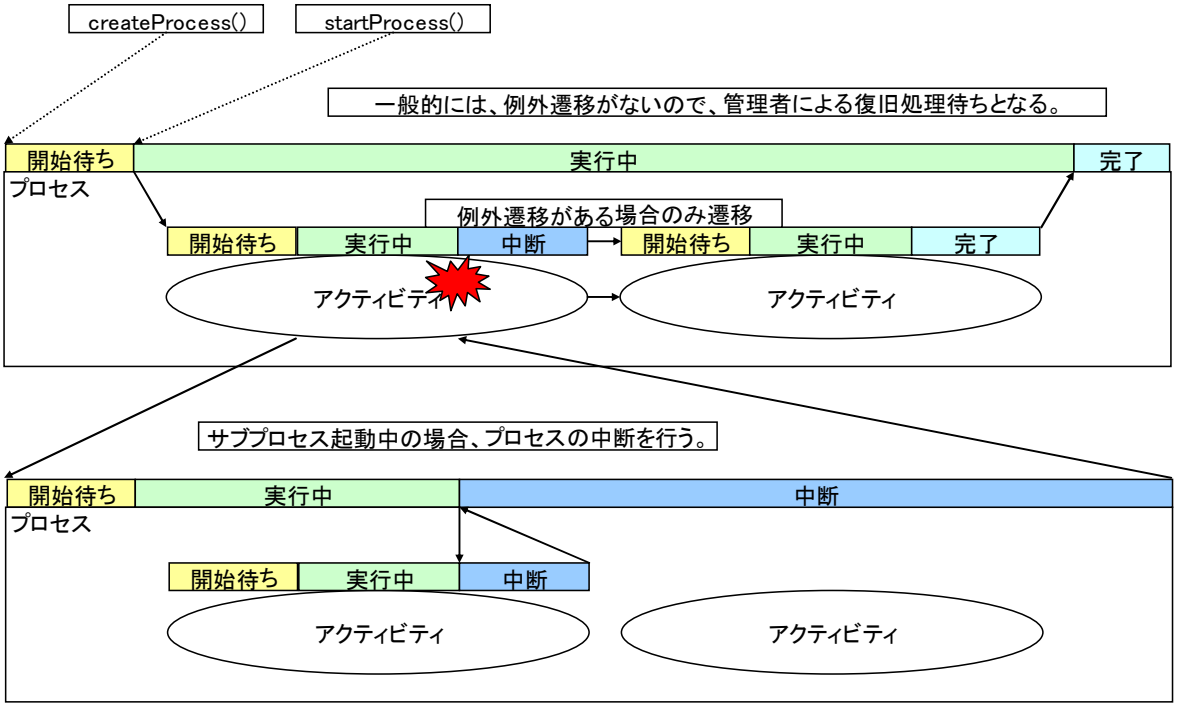
(3) 正常系状態遷移イメージ



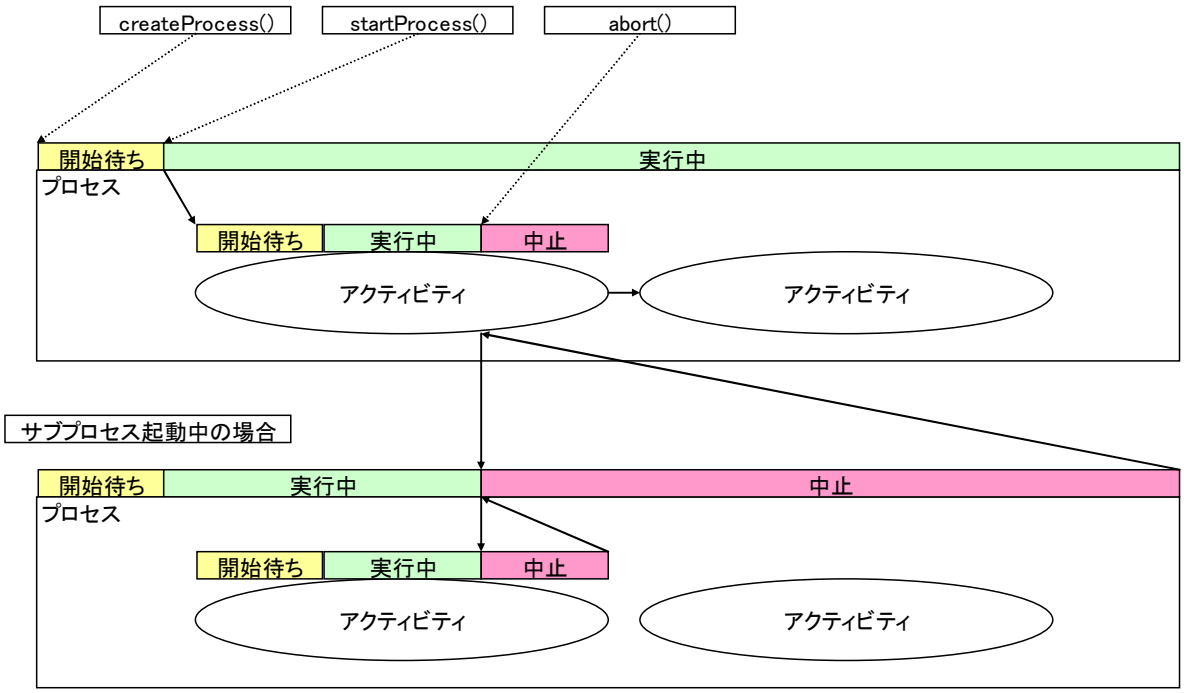
(4) 保留処理の状態遷移イメージ



(5) 異常による処理中断の状態遷移イメージ



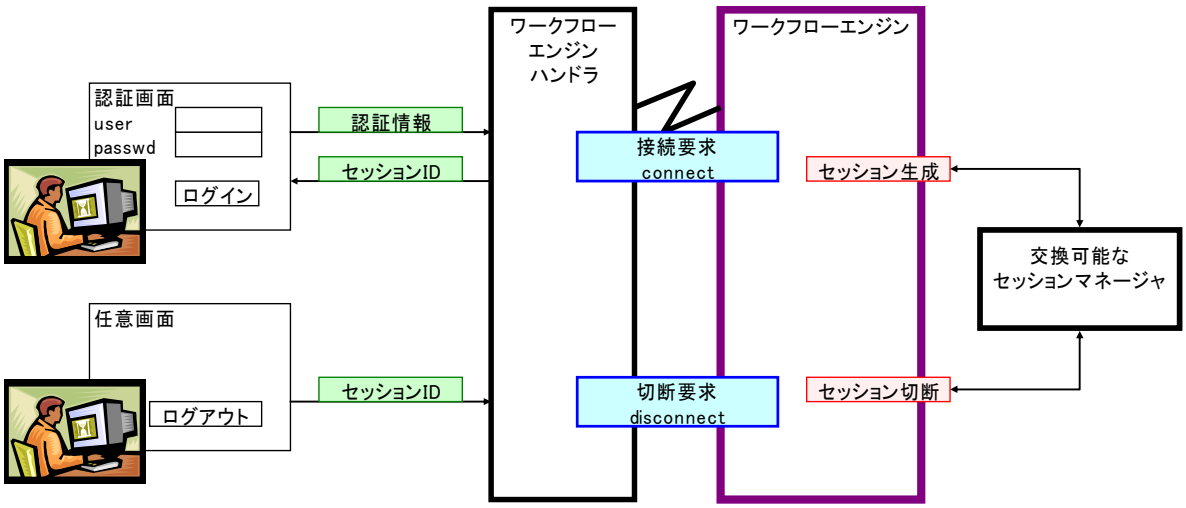
(6) 処理の中止の状態遷移イメージ



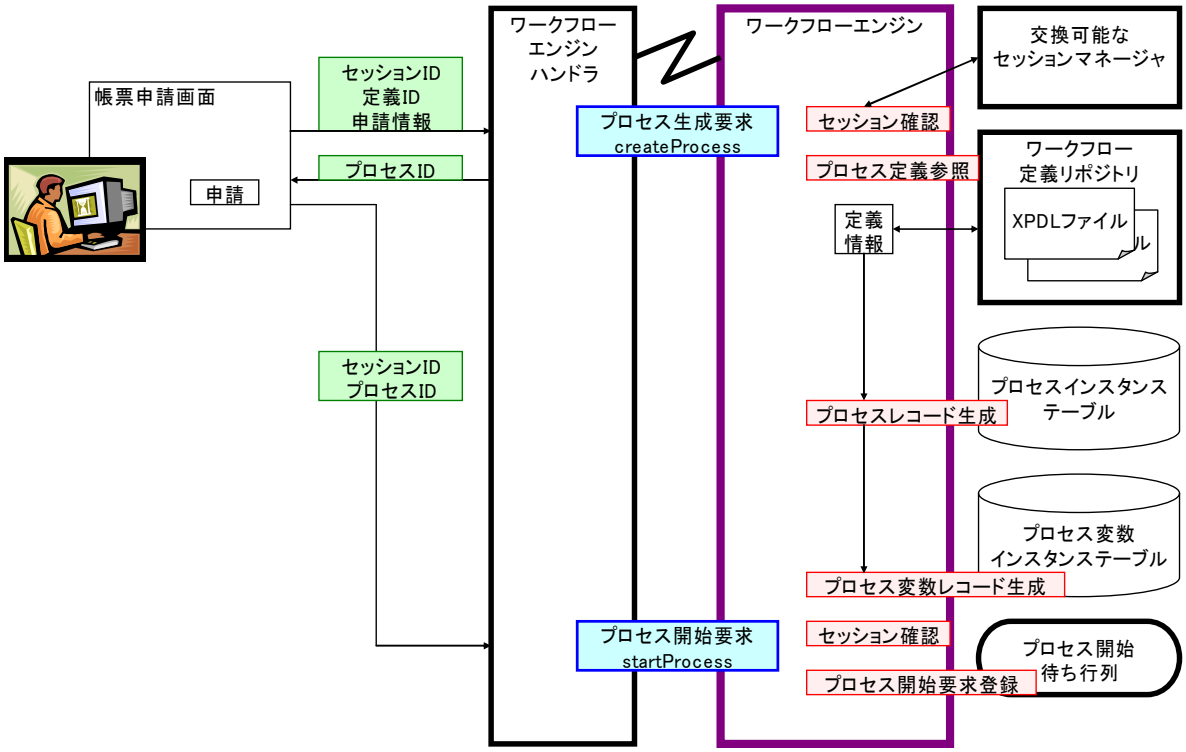
(7) 正常系処理概要

以下に、状態遷移管理機能の動作イメージを、処理要求の種類ごとに図示する。なお、図内に示すセッションマネージャなどのコンポーネントについては、「連携コンポーネント」としてまとめて後述する。

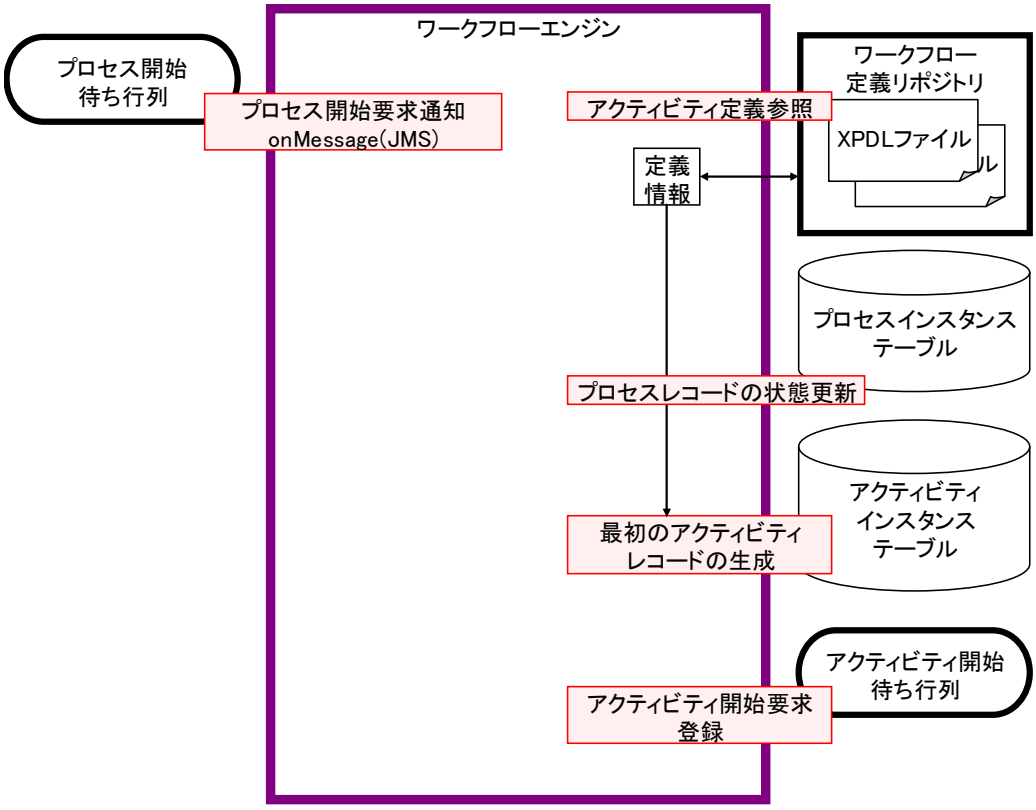
1) ワークフローエンジンへの接続と切断



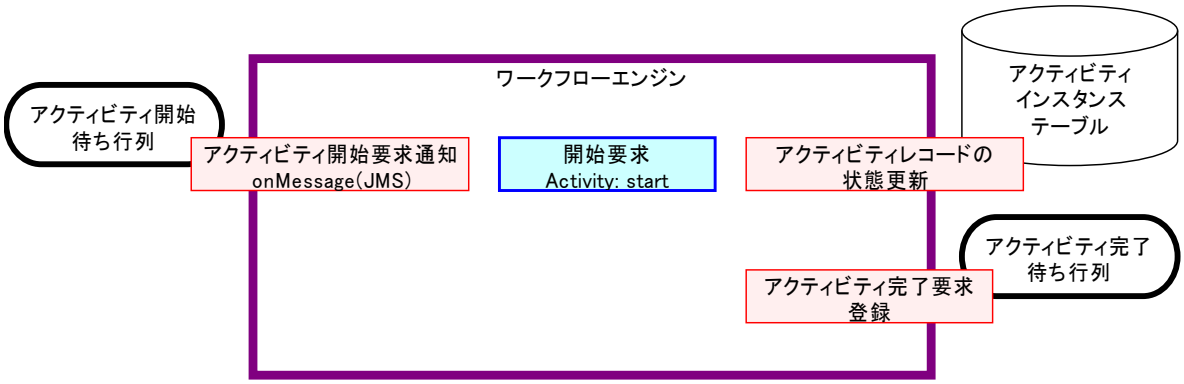
2)　ワークフロープロセスの生成



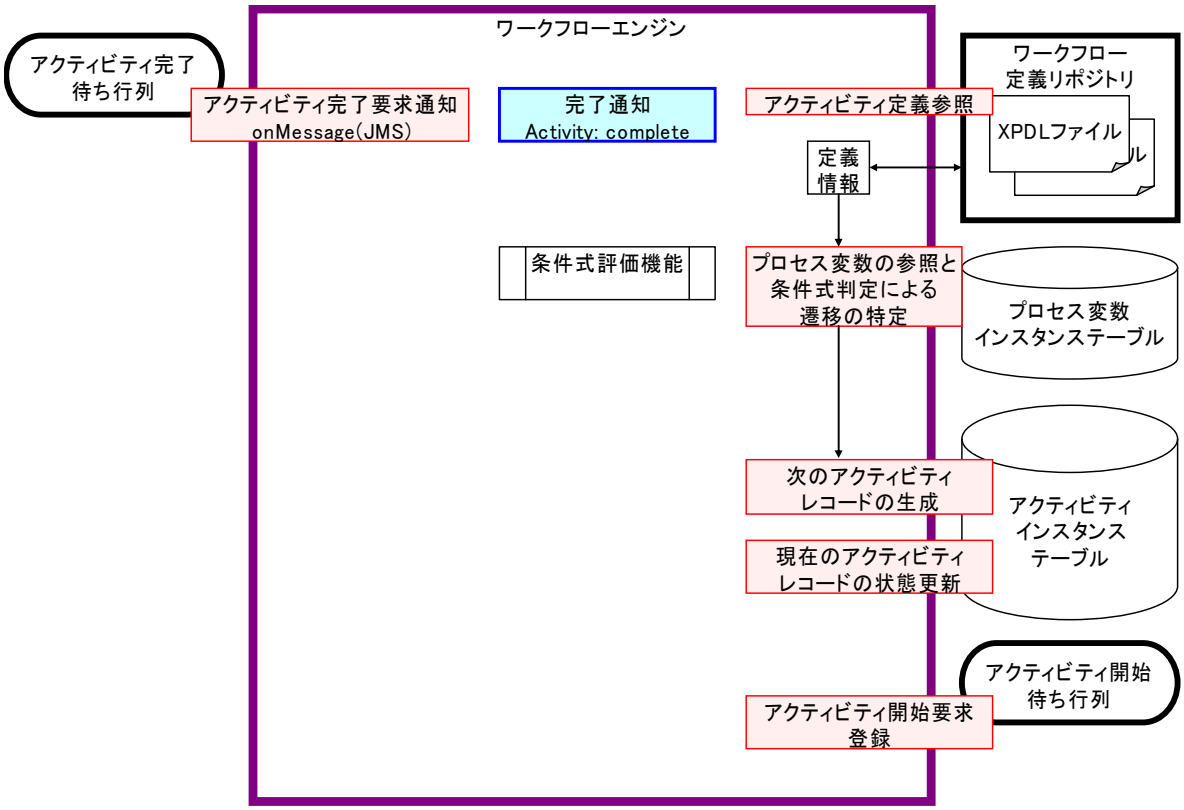
3)　ワークフロープロセスの開始



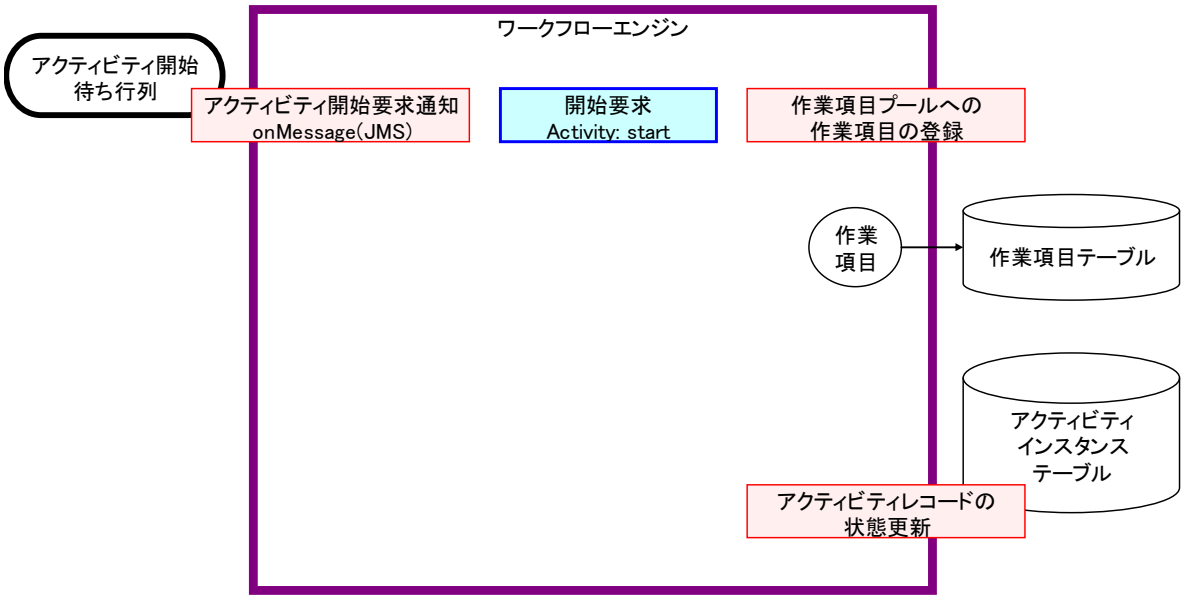
4) 開始アクティビティの開始



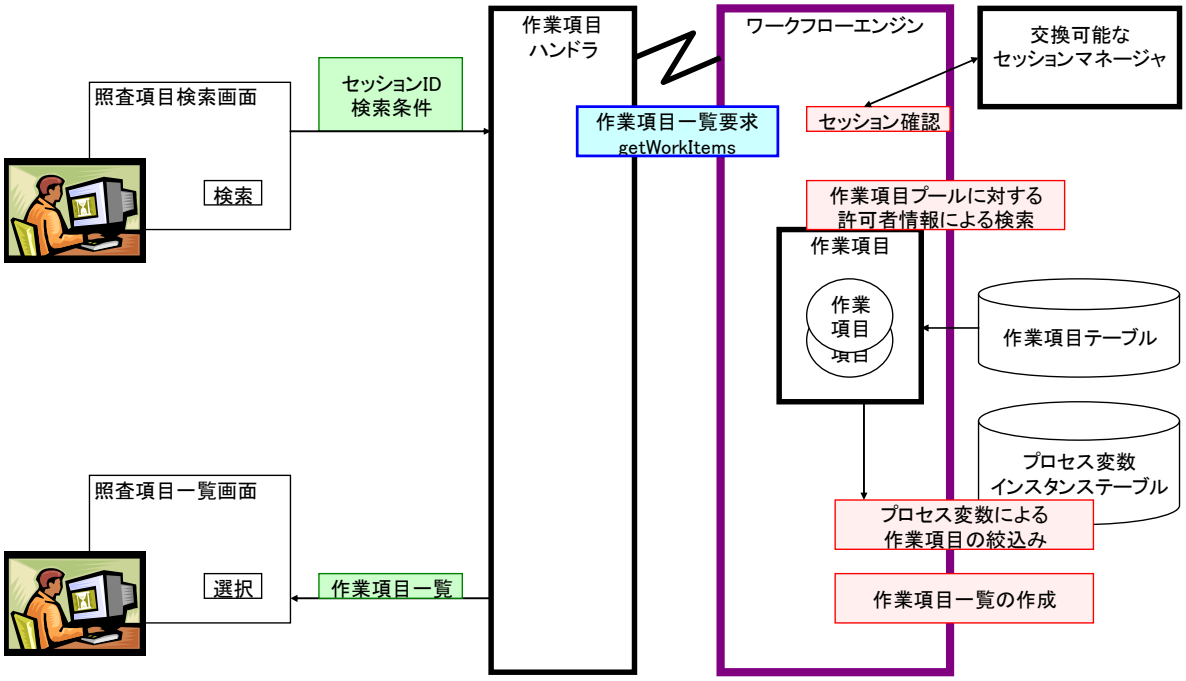
5) 開始アクティビティの完了



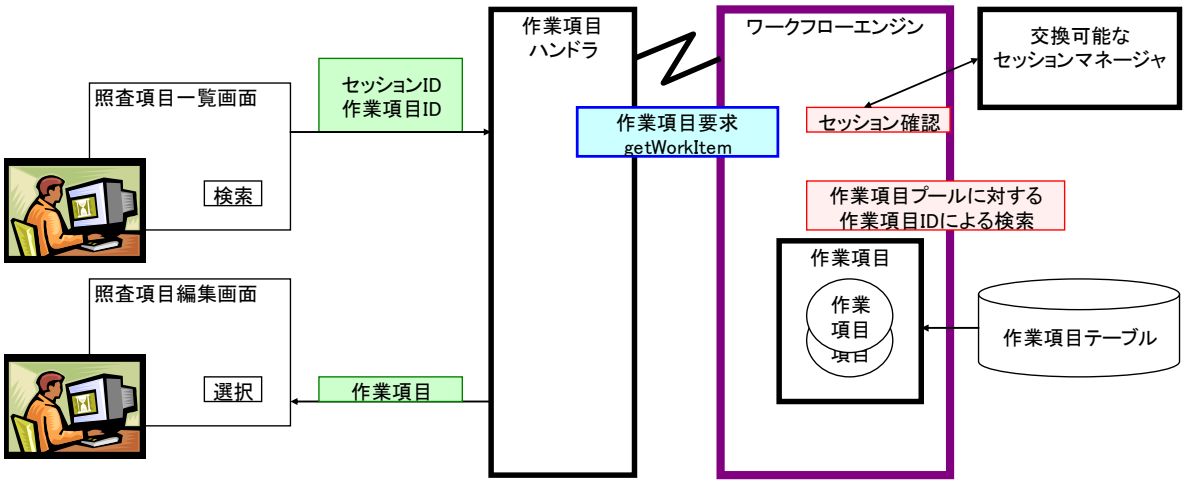
6) 対話型アクティビティの開始



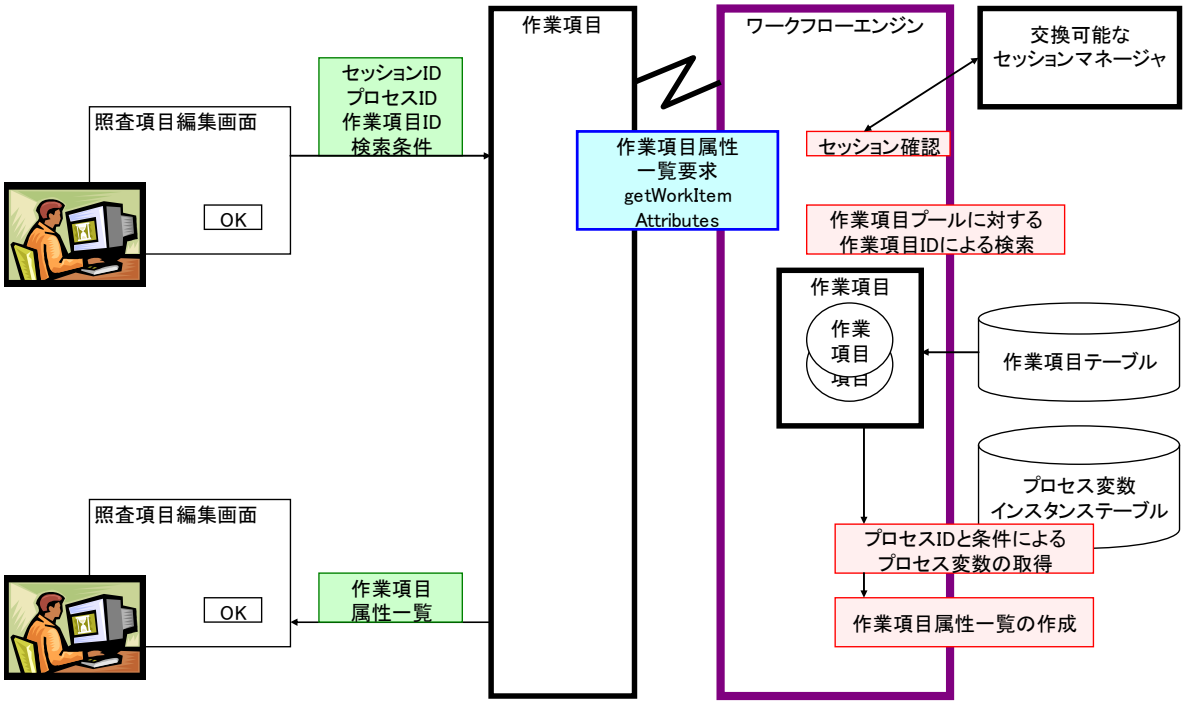
7) 作業項目一覧の取得



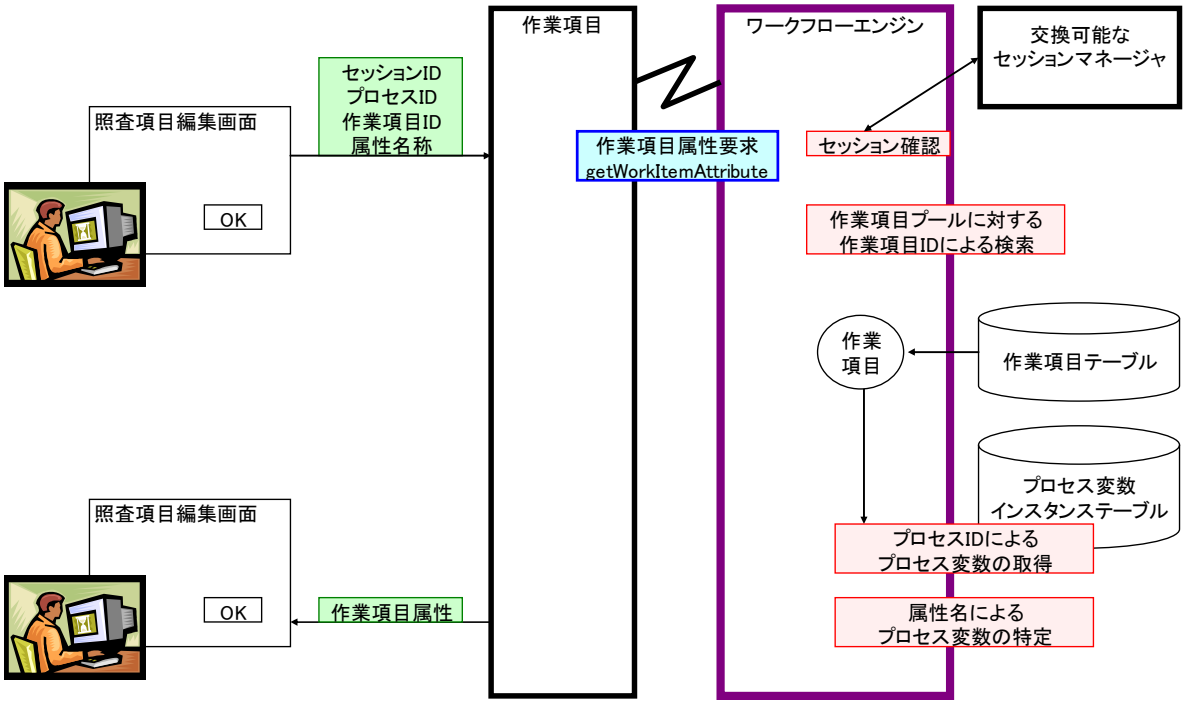
8)　作業項目の取得



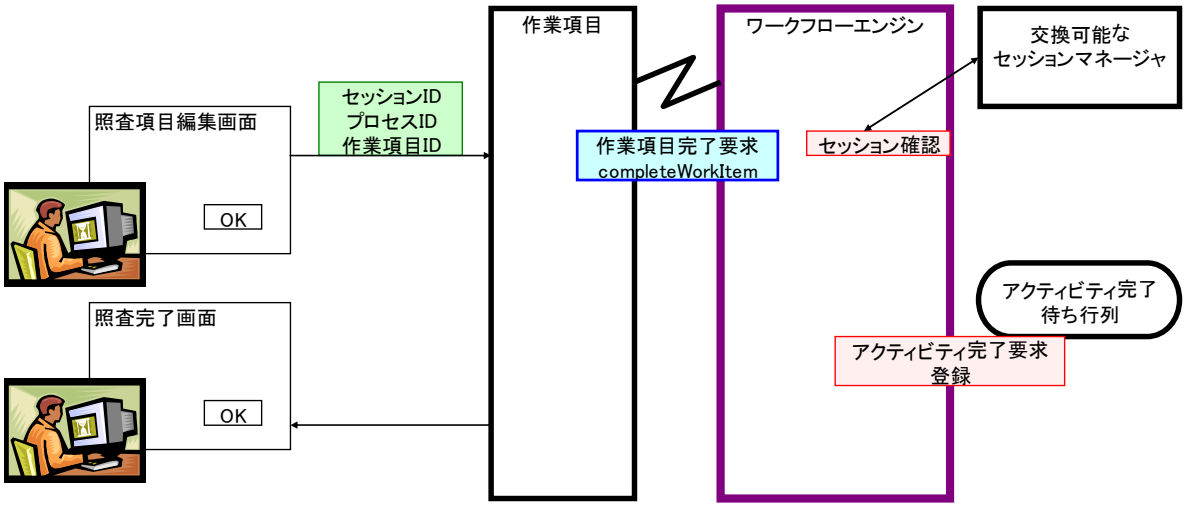
9)　作業項目属性一覧の取得



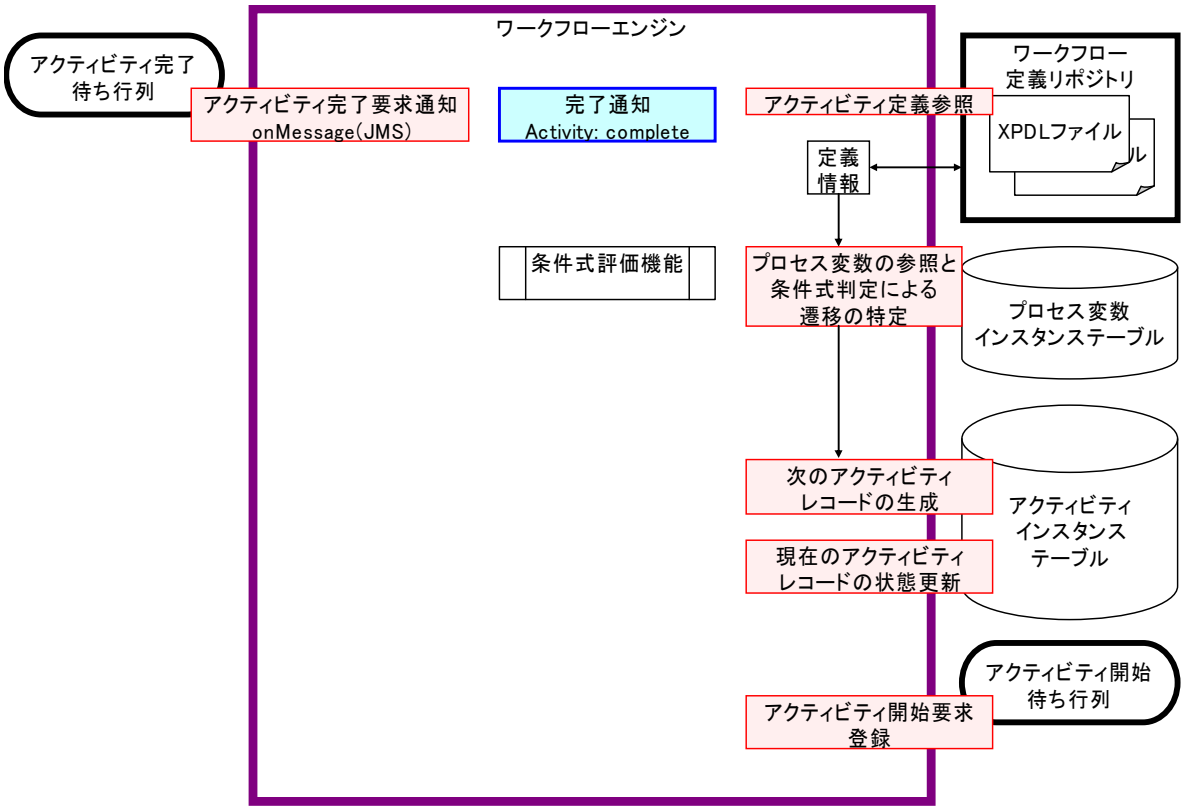
10）作業項目属性の取得



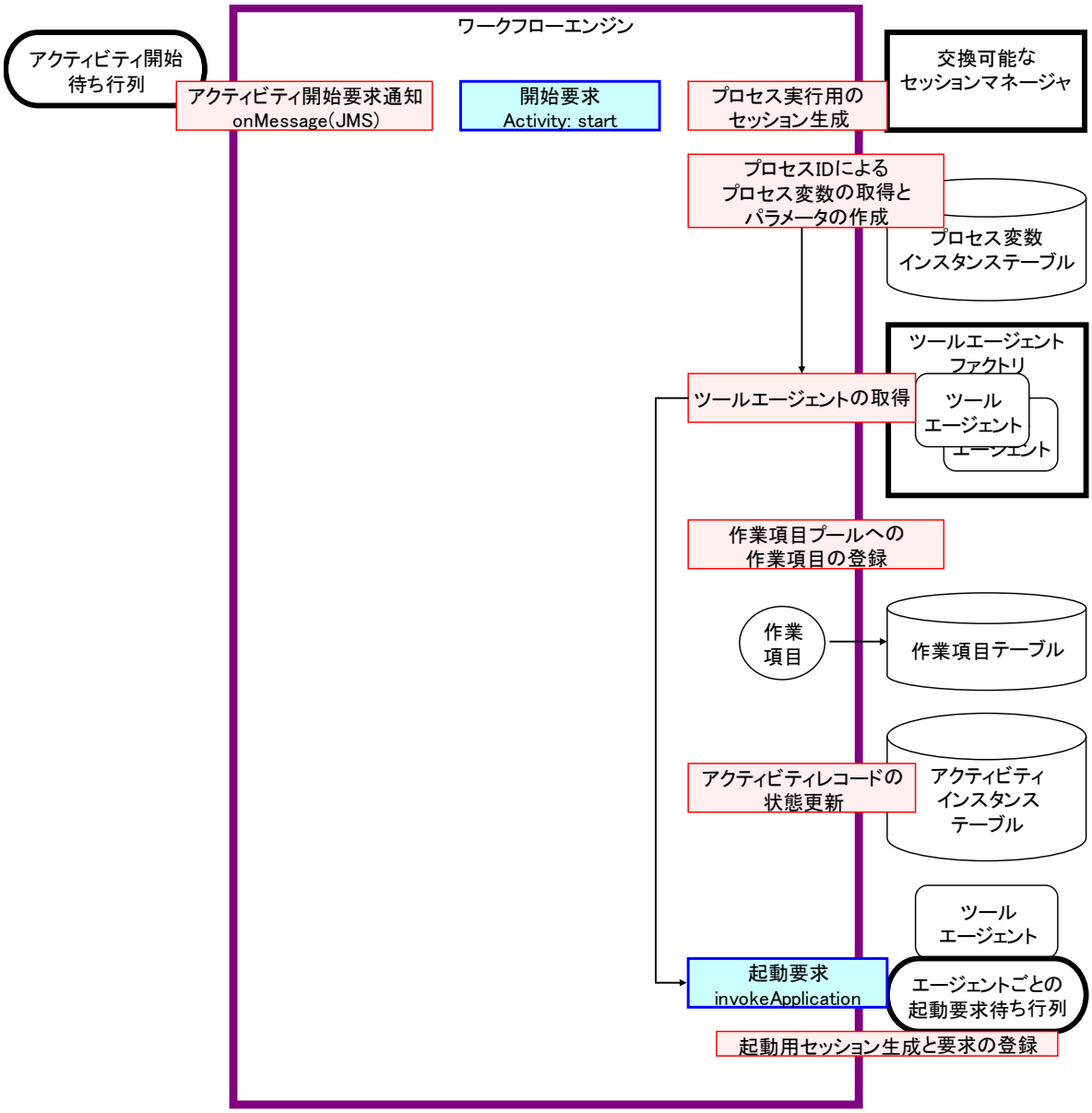
11）作業項目の完了



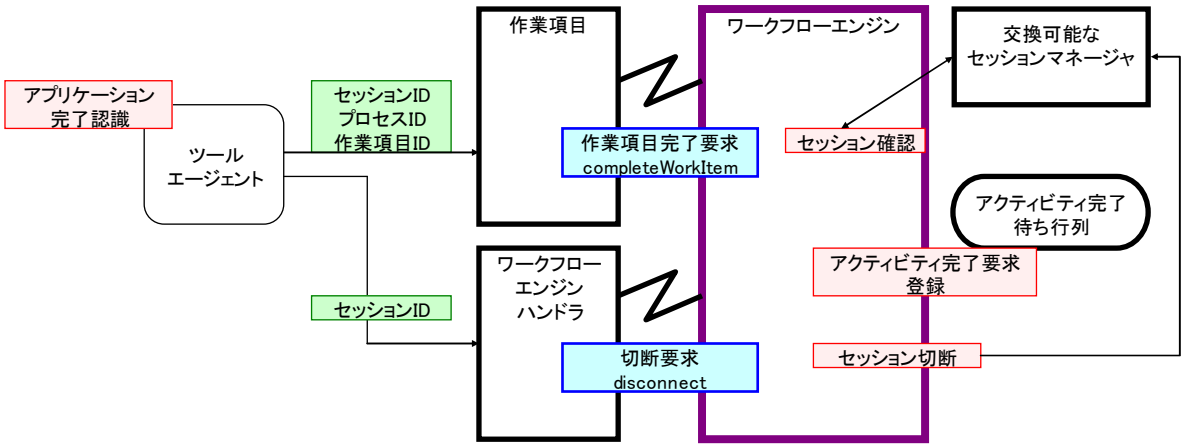
12）対話型アクティビティの完了



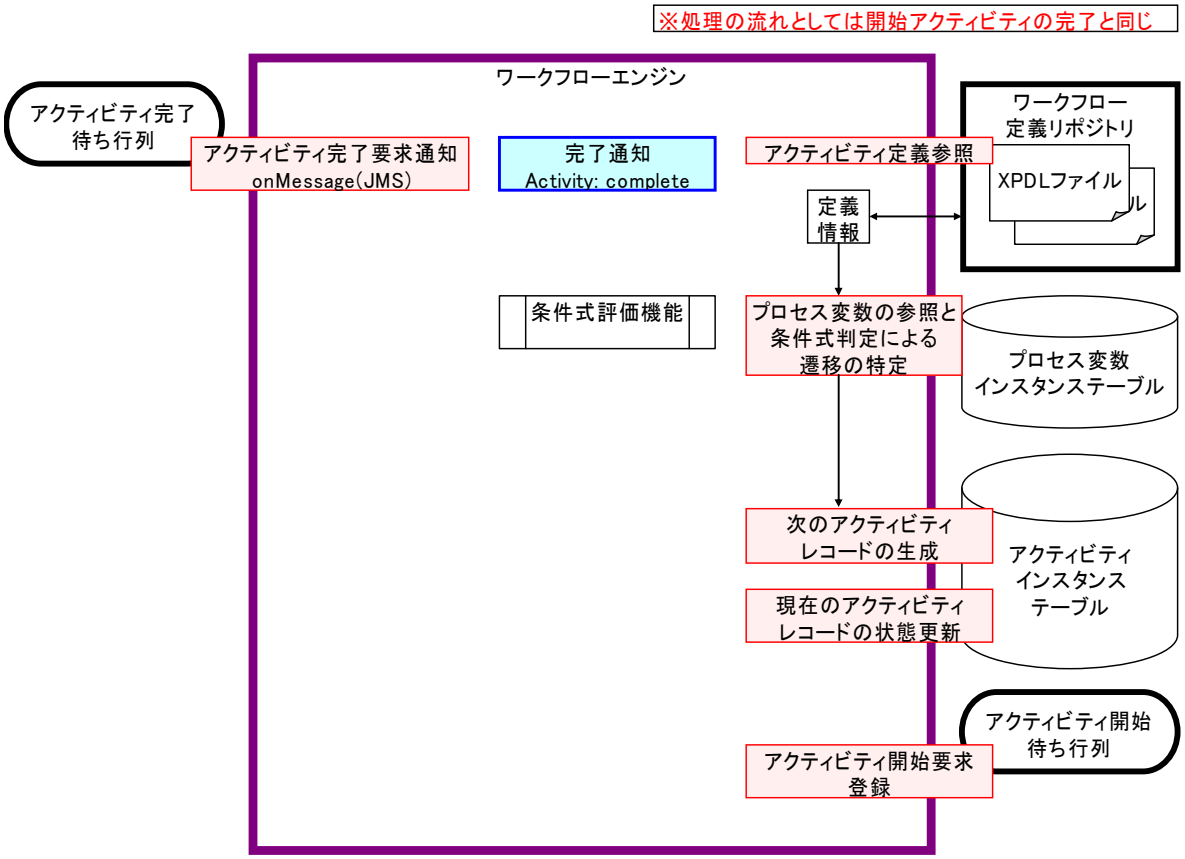
13）自動実行アクティビティの開始



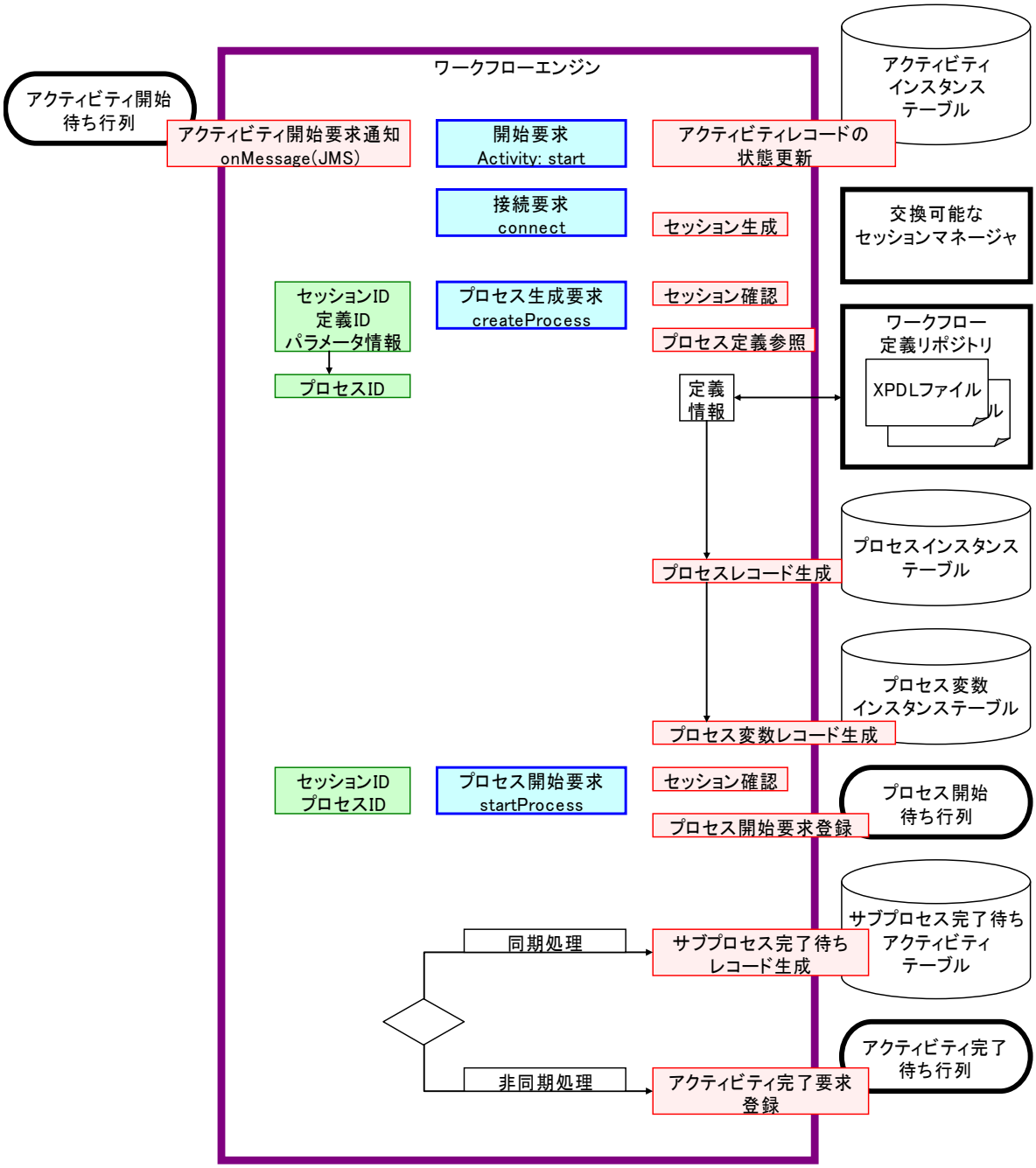
14）ツールエージェントの完了



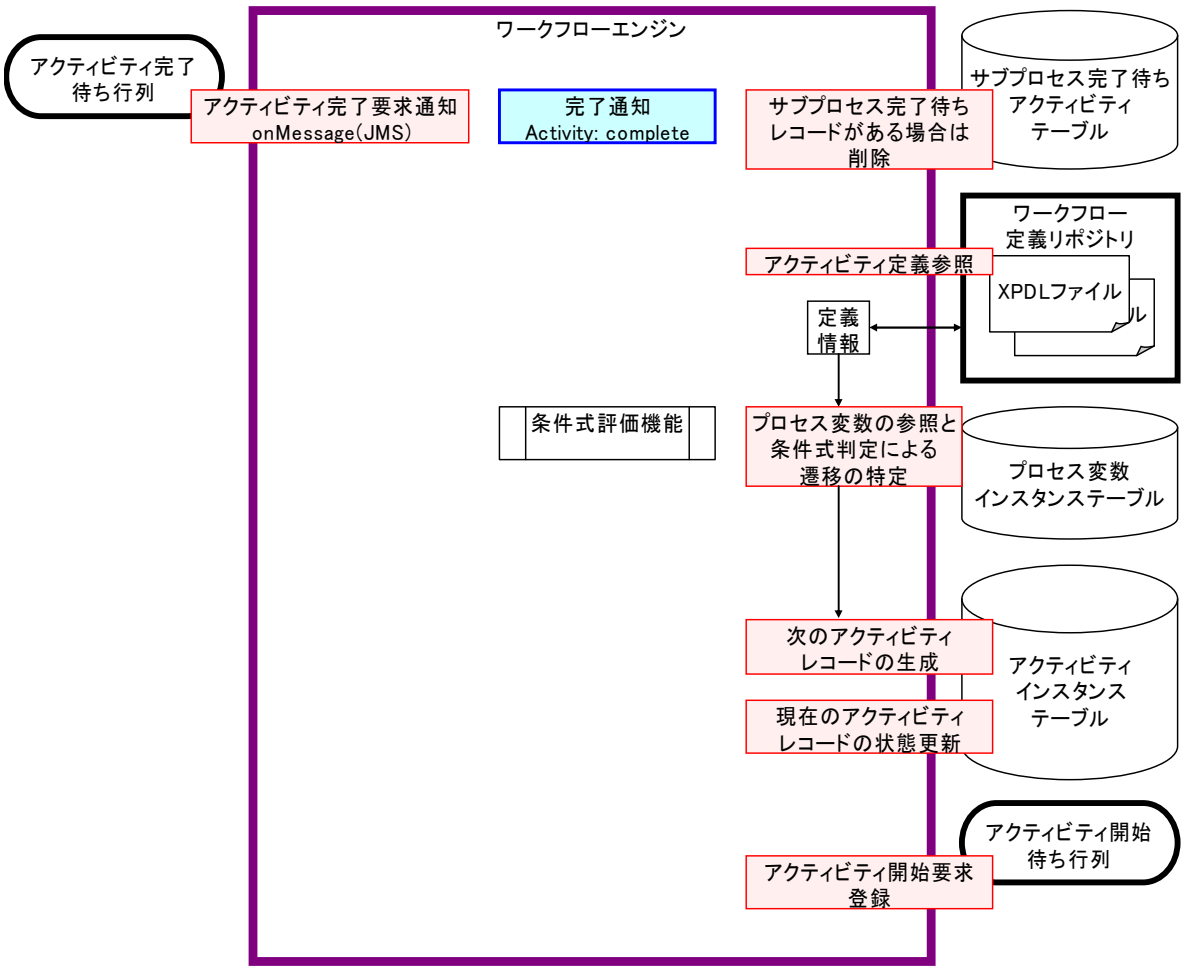
15）自動実行アクティビティの完了



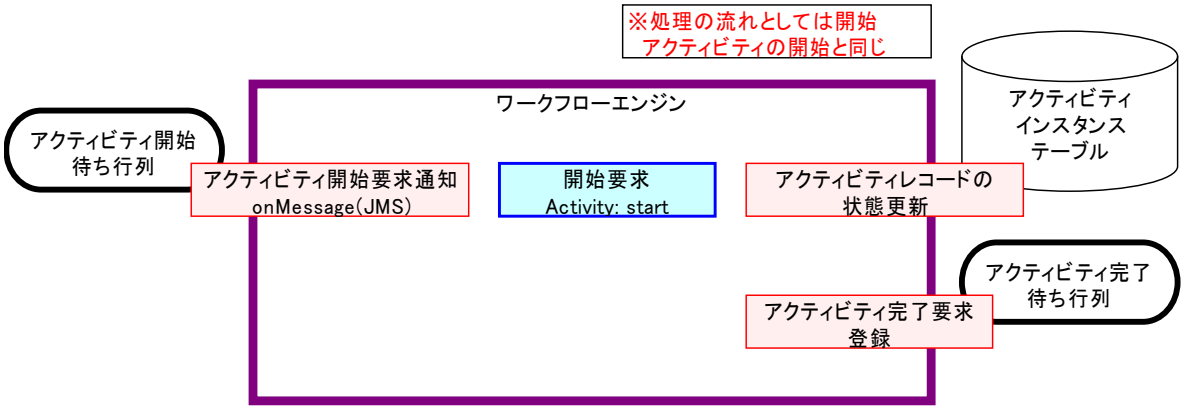
16）サブプロセス起動アクティビティの開始



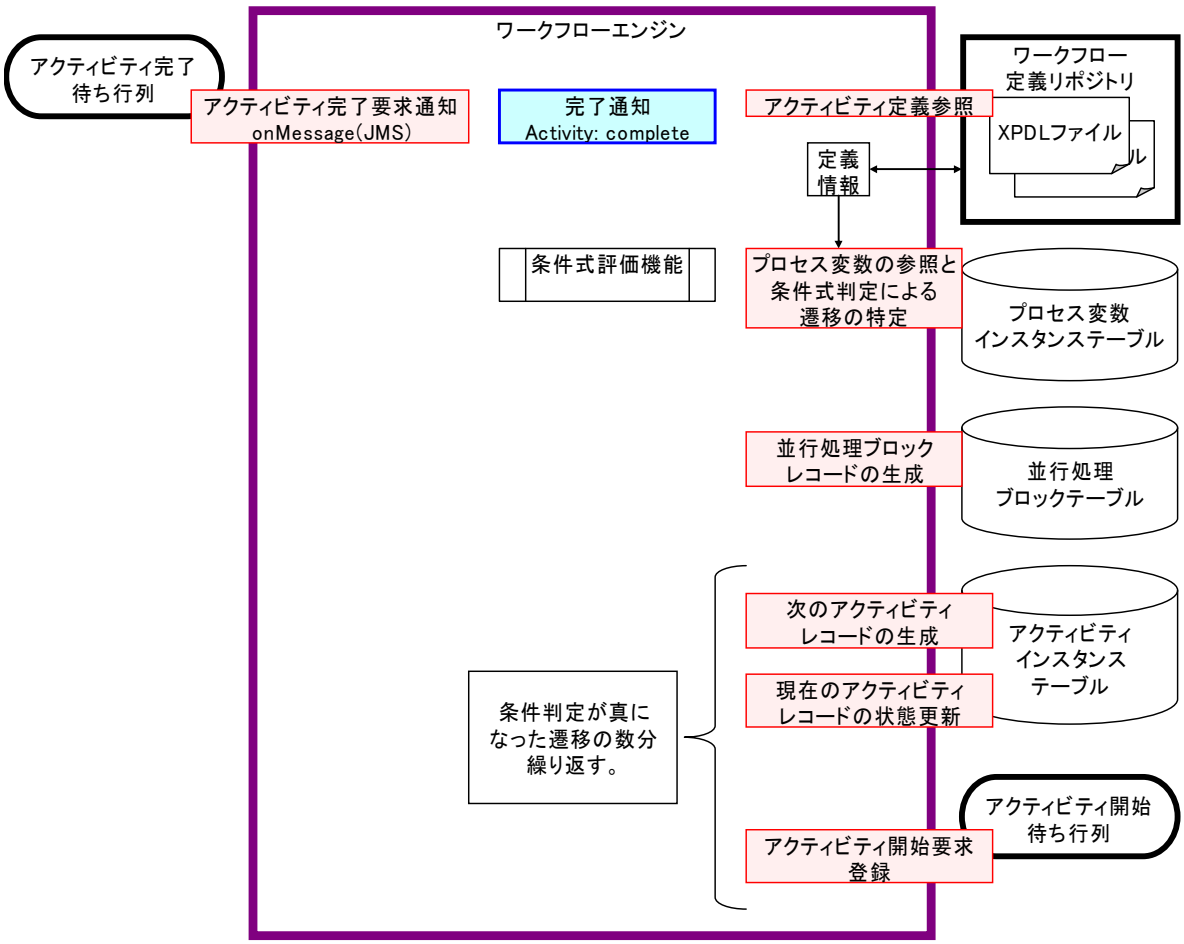
17）サブプロセス起動アクティビティの完了



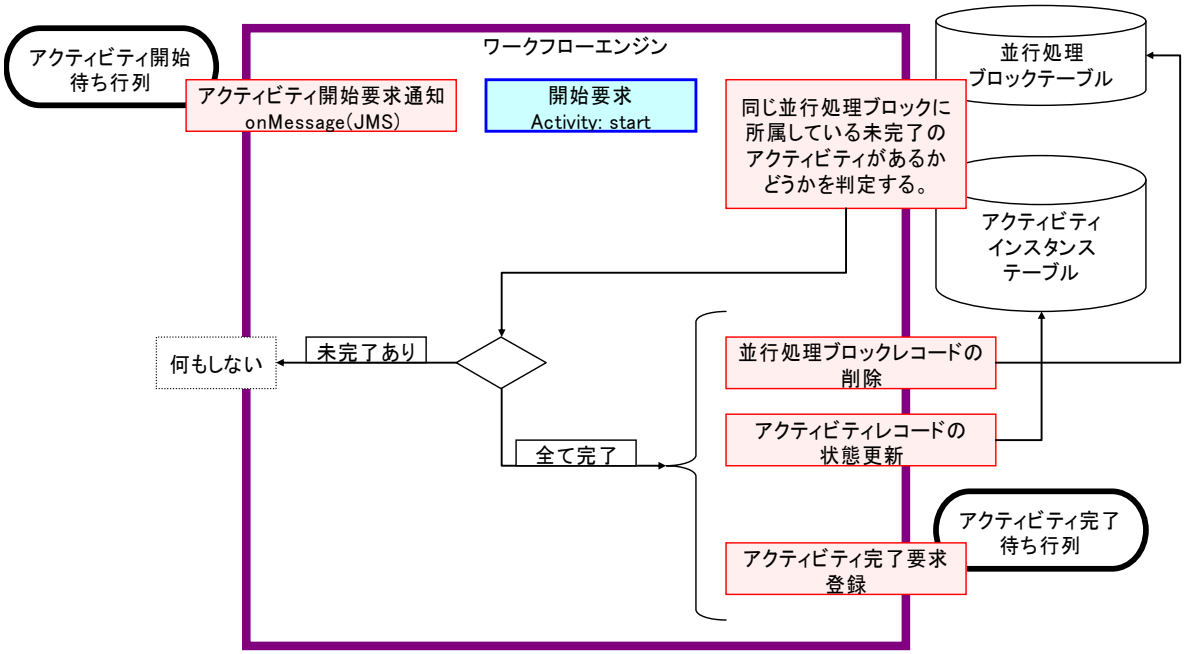
18）並行処理開始アクティビティの開始



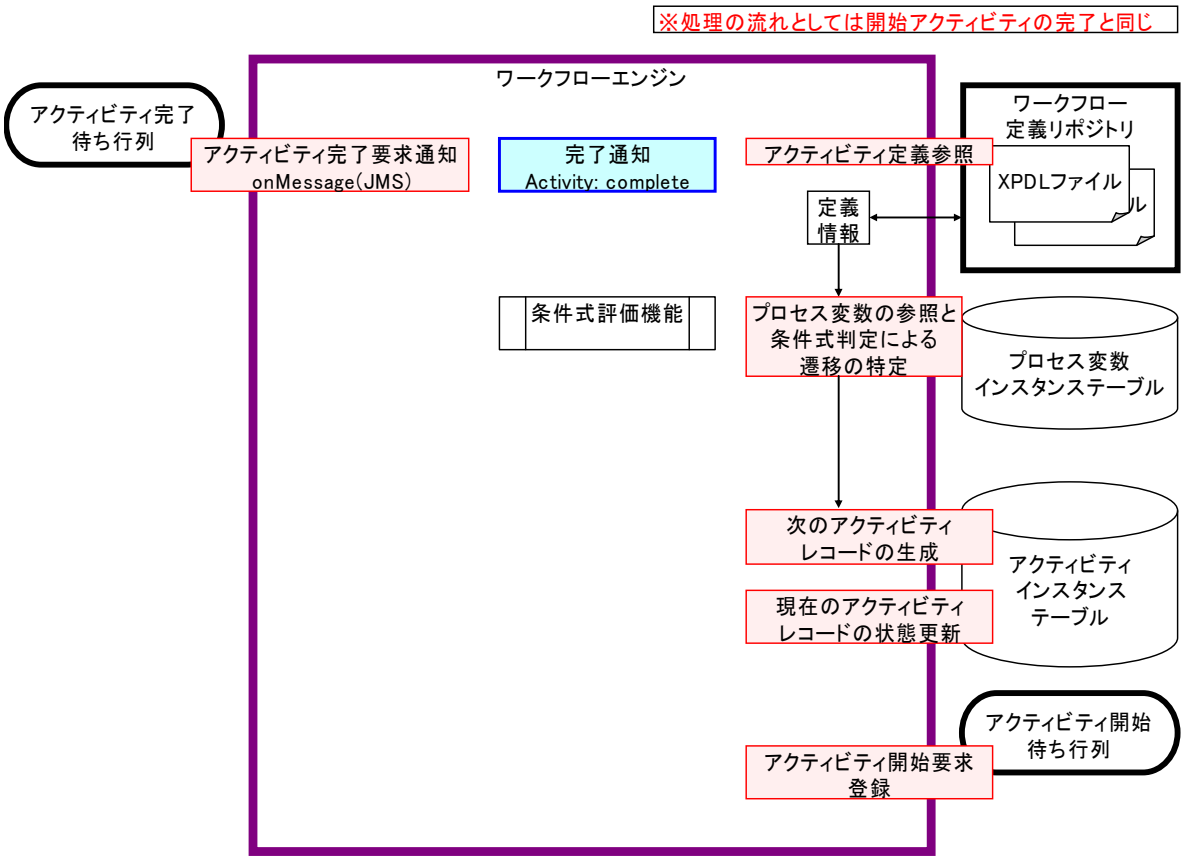
19）並行処理開始アクティビティの完了



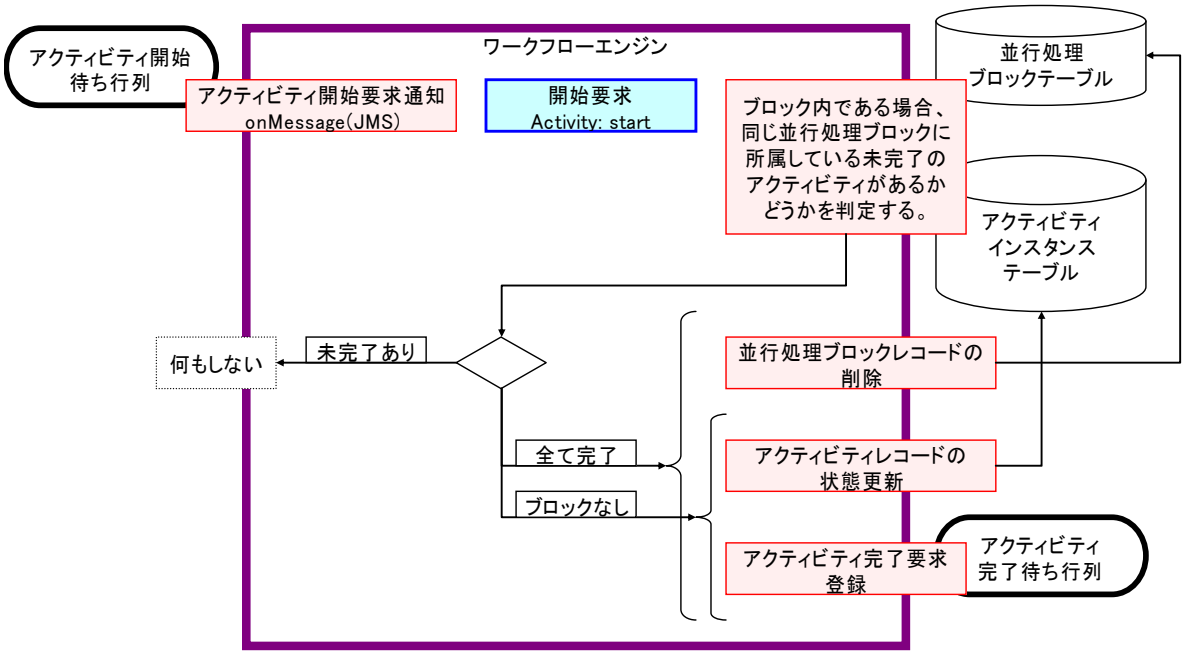
20）並行処理終了アクティビティの開始



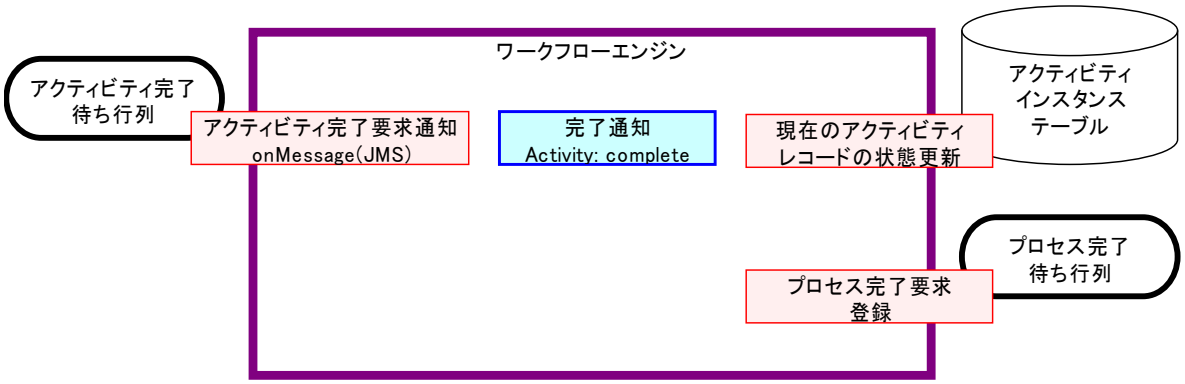
21）並行処理終了アクティビティの完了



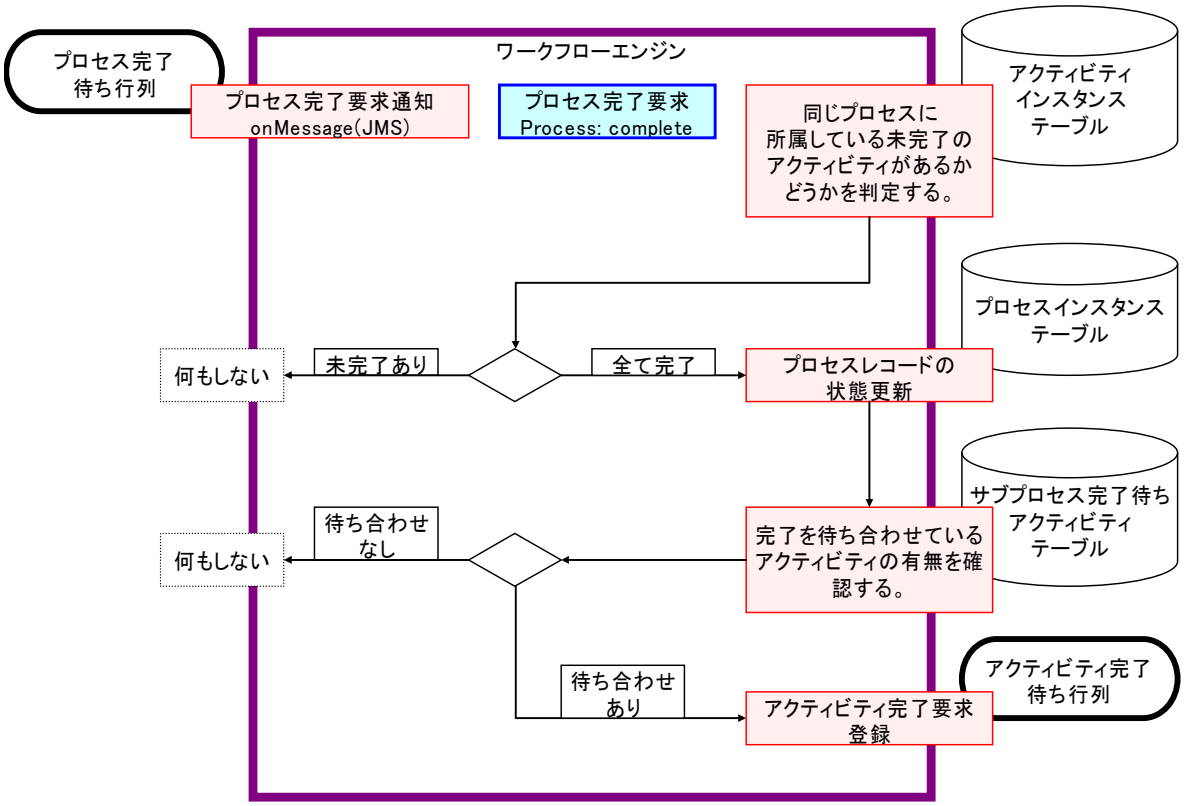
22）終了アクティビティの開始



23）終了アクティビティの完了

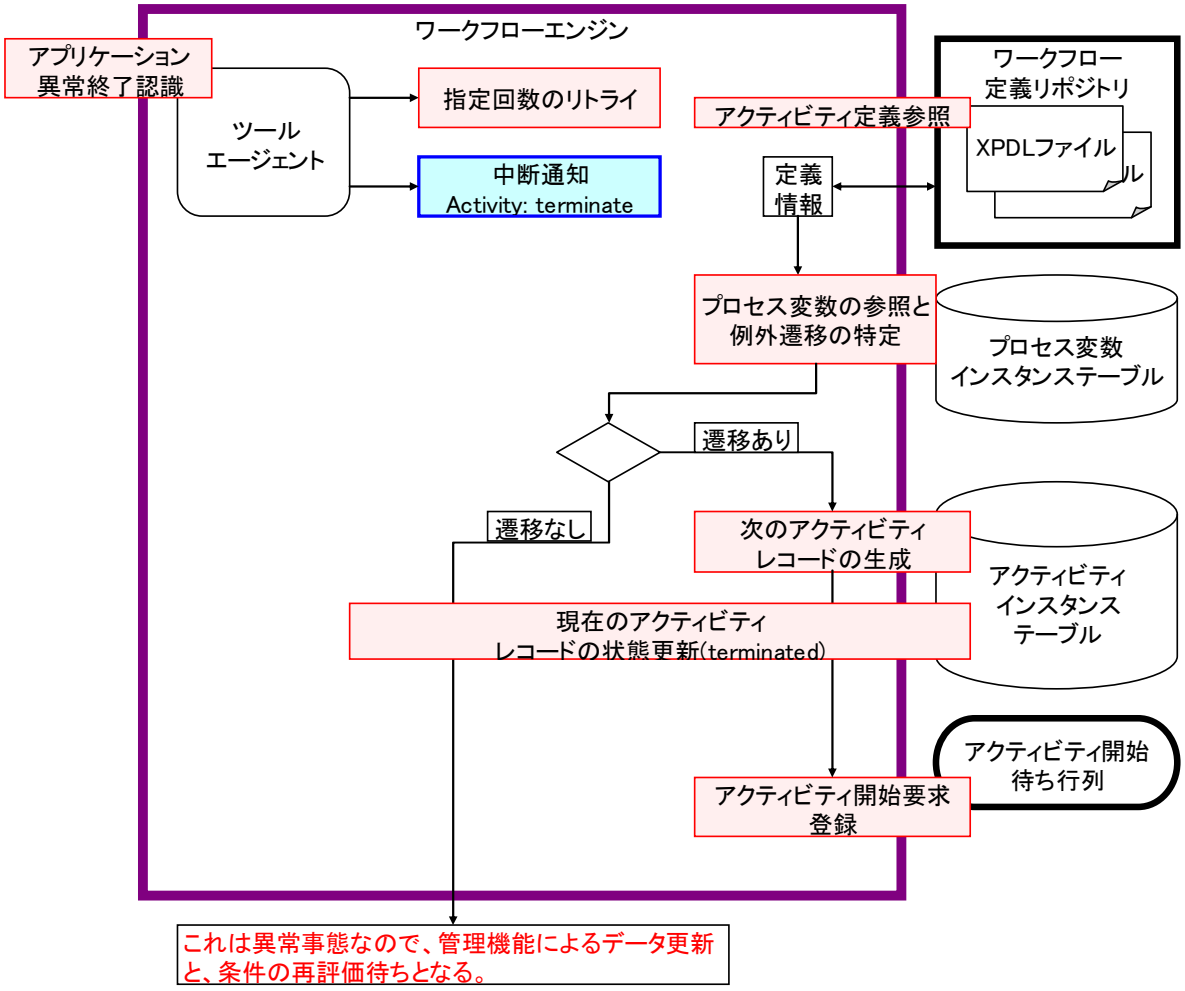


24）ワークフロープロセスの完了

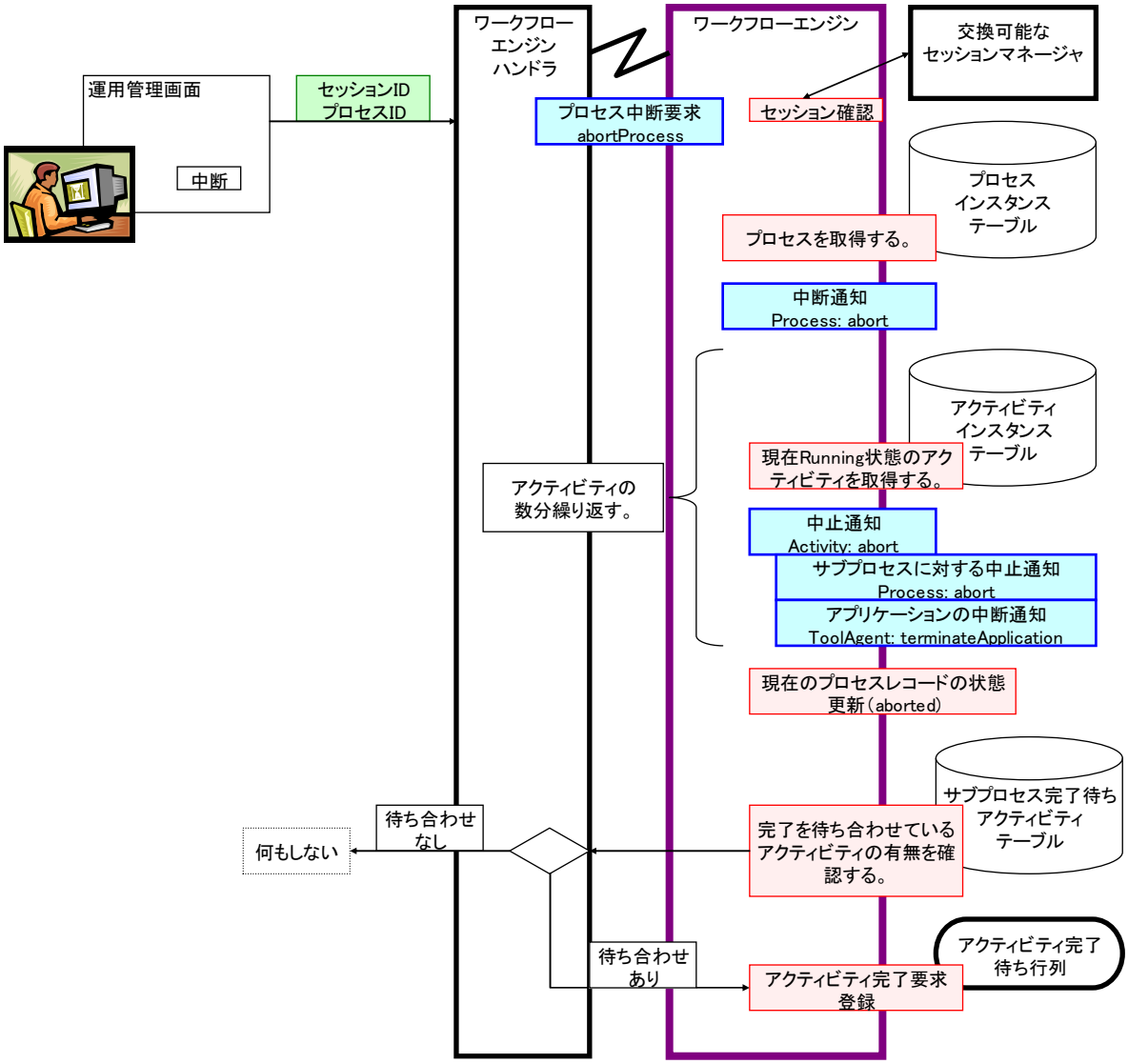


(8) 異常系処理概要

1) アプリケーションなどが異常終了した場合



2)　プロセスの中止


運用管理画面
中断
セッションID
プロセスID
ワークフロー
エンジン
ハンドラ
ワークフローエンジン
交換可能な
セッションマネージャ
プロセス中断要求
abortProcess
セッション確認
プロセス
インスタンス
テーブル
プロセスを取得する。
中断通知
Process: abort
アクティビティ
インスタンス
テーブル
現在Running状態のアク
ティビティを取得する。
アクティビティの
数分繰り返す。
中止通知
Activity: abort
サブプロセスに対する中止通知
Process: abort
アプリケーションの中断通知
ToolAgent: terminateApplication
現在のプロセスレコードの状態
更新（aborted）
サブプロセス完了待ち
アクティビティ
テーブル
待ち合わせ
なし
何もしない
完了を待ち合わせている
アクティビティの有無を確
認する。
待ち合わせ
あり
アクティビティ完了
待ち行列
アクティビティ完了要求
登録

13.1.3 アプリケーション実行制御機能

(9) 概要

ワークフローの定義に従ってアプリケーションを起動し、アクティビティの実行／終了の監視と正常／異常終了等の状態管理を行う。

アプリケーションから発生する例外を処理し、指定した回数の再実行を行う。指定回数を越えてエラーとなった場合は、アクティビティの異常終了として状態を記録する。

- スクリプトや外部アプリケーションの起動
- 起動用パラメータの引渡し
- 実行状態、終了状態の管理
- 例外処理とアクティビティの再実行

13.1.4 XPDL 定義ファイル解析機能

(1) 概要

ワークフロー定義ファイルの XML スキーマ定義を解析する機能である。ワークフローエンジンが XPDL の定義情報解析時に使用する。なお、本ワークフロー運用基盤で解析対象とするタグは、WfMC インタフェース-1 で規定する必須タグの全てと主要な任意タグであり、解析後に生成される各定義オブジェクトのインタフェースを通して定義情報にアクセスする。インタフェースについては、「3.1 ワークフロー定義」を参照のこと。

1) ワークフロー定義全体の情報（必須）

- ID（必須）
- 名前（任意）
- 備考（任意）
- XPDL のバージョン（必須）
- 定義の作成日（必須）
- 定義のバージョン（任意）
- 作成者（任意）

2) プロセスに関する情報（任意）

- ID（プロセスが定義された場合は必須）
- プロセス名（任意）
- 備考（任意）
- プロセスの責任者の ID（任意）
- 実行期限（任意）

- 実行期限の単位（年／月／日／時／分／秒）（任意）

3) アクティビティに関する情報（任意）

- ID（アクティビティが定義された場合は必須）
- アクティビティ名（任意）
- 備考（任意）
- アクティビティの種類（アクティビティが定義された場合は必須）
- 割り当てたアプリケーション ID（任意）
- 割り当てたサブプロセス ID（任意）
- 同時実行の範囲を表すブロック ID（任意）

4) 遷移に関する情報（任意）

- ID（遷移が定義された場合は必須）
- 備考（任意）
- 遷移元のアクティビティの ID（遷移が定義された場合は必須）
- 遷移先のアクティビティの ID（遷移が定義された場合は必須）

5) アプリケーションに関する情報（任意）

- ID（アプリケーションが定義された場合は必須）
- アプリケーション名（任意）
- 備考（任意）
- パラメータ（任意）

6) ワークフロー参加者に関する情報（任意）

- ID（ワークフロー参加者が定義された場合は必須）
- ワークフロー参加者の種類（ワークフロー参加者が定義された場合は必須）
- 備考（任意）

7) プロセス変数に関する情報（任意）

- ID（プロセス変数が定義された場合は必須）
- プロセス変数名（任意）
- 備考（任意）
- プロセス変数の型（プロセス変数が定義された場合は必須）
- 初期値（任意）

(2) 利用ソフトウェア

XPDL 定義ファイル解析機能を実現するために、Apache プロジェクトのオープン

ソース製品である XMLBeans 1.0 を利用する。

(3) 定義リポジトリ

定義の検索・取得は、定義リポジトリが提供するインタフェースを通じて行うものとする。

13.2 ワークフローAPI

WfMC のインタフェース２、３、４に定義される各ワークフローAPI と、本ワークフローシステム運用基盤における API の対応の概略を以下の表に示す。詳細は、資料編を参照のこと。

なお、対応作業にあたっては、以下の方針を適用した。

- リモートオブジェクトの種類を限定する。
- 戻り値には、Java のコレクション API は使用せず、配列を使用する。

No	WfMC API	マップクラス	メソッド
1	WAPI Connection Functions		
1	WMConnect()	WorkflowEngineHandler	connect()
2	WMDisconnect()	WorkflowEngineHandler	disconnect()
2	WAPI Process Control Functions		
3	WMOpenProcessDefinitionsList()	DefinitionRepository	getProcessDefinitions()
4	WMFetchProcessDefinition()	不要	
5	WMCloseProcessDefinitionsList()	不要	
6	WMOpenProcessDefinitionStatesList()	DefinitionRepository	getProcessDefinitionStates()
7	WMFetchProcessDefinitionState()	不要	
8	WMCloseProcessDefinitionStatesList()	不要	
9	WMChangeProcessDefinitionState()	DefinitionRepository	changeProcessDefinitionState()
10	WMCreateProcessInstance()	WorkflowEngineHandler	createProcess()
11	WMStartProcess()	WorkflowEngineHandler	startProcess()
12	WMTerminateProcessInstance()	WorkflowEngineHandler	terminateProcess()
13	WMOpenProcessInstanceStatesList()	WorkflowEngineHandler	getProcessStates()
14	WMFetchProcessInstanceState()	不要	
15	WMCloseProcessInstanceStatesList()	不要	
16	WMChangeProcessInstanceState()	WorkflowEngineHandler	changeProcessState()
17	WMOpenProcessInstanceAttributesList()	WorkflowEngineHandler	getProcessAttributes()
18	WMFetchProcessInstanceAttribute()	不要	
19	WMCloseProcessInstanceAttributesList()	不要	
20	WMGetProcessInstanceAttributeValue()	WorkflowEngineHandler	getProcessAttributeValue()
21	WMAssignProcessInstanceAttribute()	WorkflowEngineHandler	assignProcessAttribute()
3	WAPI Activity Control Functions		
22	WMOpenActivityInstanceStatesList()	WorkflowEngineHandler	getActivityStates()
23	WMFetchActivityInstanceState()	不要	
24	WMCloseActivityInstanceStatesList()	不要	
25	WMChangeActivityInstanceState()	WorkflowEngineHandler	changeActivityState()
26	WMOpenActivityInstanceAttributesList()	WorkflowEngineHandler	getActivityAttributes()
27	WMFetchActivityInstanceAttribute()	不要	

28	WMCloseActivityInstanceAttributesList()	不要	
29	WMGetActivityInstanceAttributeValue()	WorkflowEngineHandler	getActivityAttributeValue()
30	WMAssignActivityInstanceAttribute()	WorkflowEngineHandler	assignActivityAttribute()
4	WAPI Process Status Functions		
31	WMOpenProcessInstancesList()	WorkflowEngineHandler	getProcesses()
32	WMFetchProcessInstance()	不要	
33	WMCloseProcessInstancesList()	不要	
34	WMGetProcessInstance()	WorkflowEngineHandler	getProcess()
5	WAPI Activity Status Functions		
35	WMOpenActivityInstancesList()	WorkflowEngineHandler	getActivities()
36	WMFetchActivityInstance()	不要	
37	WMCloseActivityInstancesList()	不要	
38	WMGetActivityInstance()	WorkflowEngineHandler	getActivity()
6	WAPI Worklist Functions		
39	WMOpenWorkList()	WorkItemHandler	getWorkItems()
40	WMFetchWorkItem()	不要	
41	WMCloseWorkList()	不要	
42	WMGetWorkItem()	WorkItemHandler	getWorkItem()
43	WMCompleteWorkItem()	WorkItemHandler	completeWorkItem()
44	WMOpenWorkitemStatesList()	WorkItemHandler	getWorkitemStates()
45	WMFetchWorkitemState()	不要	
46	WMCloseWorkitemStatesList()	不要	
47	WMChangeWorkitemState()	WorkItemHandler	changeWorkitemState()
48	WMReassignWorkItem()	WorkItemHandler	reassignWorkItem()
49	WMOpenWorkItemAttributesList()	WorkItemHandler	getWorkItemAttributes()
50	WMFetchWorkItemAttribute()	不要	
51	WMCloseWorkItemAttributesList()	不要	
52	WMGetWorkItemAttributeValue()	WorkItemHandler	getWorkItemAttributeValue()
53	WMAssignWorkItemAttribute()	WorkItemHandler	assignWorkItemAttribute()
7	WAPI Administration Functions		
54	WMChangeProcessInstancesState()	WorkflowAdminHandler	changeProcessesState()
55	WMChangeActivityInstancesState()	WorkflowAdminHandler	changeActivitiesState()
56	WMTerminateProcessInstances()	WorkflowAdminHandler	terminateProcesses()
57	WMAssignProcessInstancesAttribute()	WorkflowAdminHandler	assignProcessesAttribute()
58	WMAssignActivityInstancesAttribute()	WorkflowAdminHandler	assignActivitiesAttribute()
59	WMAbortProcessInstances()	WorkflowAdminHandler	abortProcesses()
60	WMAbortProcessInstance()	WorkflowAdminHandler	abortProcess()
8	WAPI Application Invocation Functions		
61	WMTAConnect & WMTADisconnect()	不要	
62	WMTAConnect & WMTADisconnect()	不要	
63	WMTAInvokeApplication()	ToolAgent	invokeApplication()
64	WMTARequestAppStatus()	ToolAgent	requestAppStatus()
65	WMTATerminateApp()	ToolAgent	terminateApplication()
9	エンジン間連携		
66	プロセスインスタンス状態の変更	InterWorkflowConnector	changeProcessState()
67	プロセスインスタンスの生成	InterWorkflowConnector	createProcess()
68	プロセスインスタンス属性の取得	InterWorkflowConnector	getProcessAttributes()
69	プロセスインスタンス状態の取得	InterWorkflowConnector	getProcessState()
70	プロセスインスタンス属性の変更通知	InterWorkflowConnector	processAttributesChanged()
71	プロセスインスタンス状態の変更通知	InterWorkflowConnector	processStateChanged()

72	プロセスインスタンス属性の設定	InterWorkflowConnector	setProcessAttributes()
72	アクティビティに対するトリガー	InterWorkflowConnector	triggerActivity()
73	プロセスインスタンス一覧の取得	InterWorkflowConnector	getProcessIDs()
74	プロセスインスタンスの放棄	InterWorkflowConnector	relinquishProcess()

13.2.1 運用管理用 API

(1) 概要

ワークフロー運用管理ツールがワークフローエンジンの情報取得、管理に使用する API である。WfMC インタフェース-２、３、及び５に機能的に準拠する。今回提供する API は以下のものである。

- プロセス状態の変更
- アクティビティ状態の変更
- プロセスの強制終了
- プロセスへの属性 の割り当て
- 全プロセス属性の取得
- アクティビティを指定した属性の割り当て
- 全プロセスの停止
- プロセスの停止

13.2.2 クライアント用 API

(1) 概要

ワークフローエンジンの機能を呼び出すクライアントアプリケーションが利用する API。WfMC インタフェース-２、３に機能的に準拠する。今回提供する API は以下のものである。

- プロセス定義の取得
- プロセスの生成
- プロセスの開始
- プロセスの強制終了
- プロセスの状態の取得
- プロセスの取得
- プロセスの状態の設定
- アクティビティの取得
- アクティビティの状態の取得
- アクティビティの状態の変更
- アクティビティの属性の取得
- アクティビティの属性の割り当て
- 作業項目 一覧の取得

- 作業項目の取得
- 作業項目の完了通知
- 作業項目の解放通知
- 作業項目の属性の取得
- 作業項目の属性の割り当て

13.2.3 アプリケーション連携用 API

(1) 概要

ワークフローエンジンから外部アプリケーションの起動および連携をおこなうための API。WfMC インタフェース-３に機能的に準拠する。今回提供する API は以下のものである。

- アプリケーション機能への接続と切断
- アプリケーションの起動
- アプリケーションの状態の取得
- アプリケーションの強制終了

13.2.4 ワークフローエンジン間連携用 API

(1) 概要

複数のワークフローエンジン間で連携を行うための API。WfMC インタフェース-４に機能的に準拠する。今回提供する API は以下のものである。

- プロセス状態の変更
- プロセスの生成
- プロセス変数の取得
- プロセスの状態の取得
- プロセスの状態の変更
- プロセス状態変更通知
- プロセス変数の設定
- プロセス一覧の取得
- プロセスの放棄

13.3 ワークフロー定義ツール

(1) 概要

ワークフロー定義を GUI ベースで作成、登録を行うためのツールである。ワークフロー定義ツールは、以下の機能を実現する。

- 連携するアプリケーションの定義
- プロセス変数の定義

- ワークフロー参加者の定義
- プロセスの定義
- アクティビティの定義
- 遷移の定義
- 定義内容の検証
- XPDL 形式でのファイル入出力
- 定義の印刷

ワークフロー定義ツールの設計は、別途、取扱説明書で代替する。

13.4 ワークフロー運用管理ツール

(1) 概要

ワークフローエンジンの運用管理を GUI ベースで行うツールである。ワークフロー運用管理ツールは、以下に示す機能を提供する。

- ネットワーク上のワークフローエンジンへの接続
- ワークフローエンジンが管理するプロセス情報の取得
- ワークフロー内のアクティビティ進行状態の表示
- プロセス変数の表示と更新
- フローの途中で保留状態となっているアクティビティの遷移条件の再評価
- 実行中のプロセスの強制終了

ワークフロー運用管理ツールの設計は、別途、取扱説明書で代替する。

13.5 通信インタフェース

13.5.1 RMI over IIOP

(1) 概要

ワークフローエンジンの各 API を Java のリモートメソッド呼び出し(RMI)から利用できるインタフェースを提供する。想定するクライアントは、LAN 内で動作する通常アプリケーションである。クライアントにはワークフロー運用管理ツールも含む。

(2) 利用ソフトウェア

Java 標準の RMI および CORBA サービスを利用する。ただし、ネームサービスについては、単体使用の場合は J2SE 添付の COS ネームサービスを利用し、Tomcat と併用する場合は Tomcat のネームサービスを利用するものとする。

13.5.2 Web サービス

(1) 概要

ワークフローエンジンの各 API を SOAP ベースの Web サービスとして利用できる

インタフェースを提供する。想定するクライアントは、インターネットも含めたネットワーク上の SOAP ベースのアプリケーションである。

(2) 利用ソフトウェア

本ワークフローシステム運用基盤と Web サービスを結びつけるために、Apache プロジェクトのオープンソース製品である Axis 2.0 を利用することとする。Axis の利用は、サーバ側とクライアント側の両方で必要である。また、Axis を動作させるアプリケーションサーバとして、Jakarta プロジェクトの Tomcat 5.0 を利用する。

Web サービスで渡すことのできるオブジェクトには制限があるので、なるべくその制限を回避した形で機能を提供する。特に Java のコレクションは、Axis における制限事項の一つなので、詳細設計では考慮が必要である。

13.5.3 Servlet

(1) 概要

ワークフローエンジンの各 API 呼び出し部分を実装した Servlet を提供する。想定するクライアントは、Web ブラウザである。Servlet で取得した情報は、HTML 形式への変換部分を個別に開発し、Web ブラウザから利用するものとする。

(2) 利用ソフトウェア

本ワークフローシステム運用基盤と Servlet エンジンを結びつけるための Servlet エンジンとして、Jakarta プロジェクトの Tomcat 5.0 を利用する。

13.6 サンプルアプリケーション

(1) 概要

ワークフローシステム運用基盤を使用したサンプルアプリケーションを提供する。サンプルアプリケーションは、文書承認フローを模した小規模システムとして提供し、文書申請用、文書承認用、承認状況通知用アプリケーションおよび承認用のフロー定義から構成する。文書承認用アプリケーションは承認権限を２つ（部長・課長）用意し、それぞれの権限者による承認作業を行う。また、承認／否認の状況は、ワークフロー定義に従って起動する承認状況通知アプリケーションの出力内容で確認を行う。

本プロジェクトで開発したワークフローエンジン及び API の機能検証は本サンプルアプリケーションを使用して行うことから、サンプルアプリケーションの設計は、本書には記載しない。

III 入出力データ編

1　ワークフロー定義仕様

1.1　XPDL スキーマ仕様

以下に、WfMC が提供する XPDL 用の XML スキーマ定義を示す。オリジナルのスキーマ定義ファイルは、WfMC の公式サイトである http:///www.wfmc.org で入手可能である。なお、本ワークフローシステム運用基盤では、これらのタグのうち、必要なもののみを使用する。赤字は、今回採用したタグを示す。青字のタグは、今回使用していないものである。

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace=http://www.wfmc.org/2002/XPDL1.0
 xmlns:xpdl=http://www.wfmc.org/2002/XPDL1.0
 xmlns:xsd=http://www.w3.org/2001/XMLSchema
 elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xsd:element name="Activities">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Activity" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Activity">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:Limit" minOccurs="0"/>
            <xsd:choice>
               <xsd:element ref="xpdl:Route"/>
               <xsd:element ref="xpdl:Implementation"/>
               <xsd:element ref="xpdl:BlockActivity"/>
            </xsd:choice>
            <xsd:element ref="xpdl:Performer" minOccurs="0"/>
            <xsd:element ref="xpdl:StartMode" minOccurs="0"/>
            <xsd:element ref="xpdl:FinishMode" minOccurs="0"/>
            <xsd:element ref="xpdl:Priority" minOccurs="0"/>
            <xsd:element ref="xpdl:Deadline" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element ref="xpdl:SimulationInformation" minOccurs="0"/>
            <xsd:element ref="xpdl:Icon" minOccurs="0"/>
            <xsd:element ref="xpdl:Documentation" minOccurs="0"/>
            <xsd:element ref="xpdl:TransitionRestrictions" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="Name" type="xsd:string"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ActivitySet">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Activities" minOccurs="0"/>
            <xsd:element ref="xpdl:Transitions" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
      </xsd:complexType>
   </xsd:element>
```

```
<xsd:element name="ActivitySets">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref="xpdl:ActivitySet" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<xsd:element name="ActualParameter" type="xsd:string"/>
<xsd:element name="ActualParameters">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref="xpdl:ActualParameter" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<xsd:element name="Application">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref="xpdl:Description" minOccurs="0"/>
         <xsd:choice>
            <xsd:element ref="xpdl:FormalParameters"/>
            <xsd:element ref="xpdl:ExternalReference" minOccurs="0"/>
         </xsd:choice>
         <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
      <xsd:attribute name="Name" type="xsd:string"/>
   </xsd:complexType>
</xsd:element>
<xsd:element name="Applications">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref="xpdl:Application" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<xsd:element name="ArrayType">
   <xsd:complexType>
      <xsd:group ref="xpdl:DataTypes"/>
      <xsd:attribute name="LowerIndex" type="xsd:NMTOKEN" use="required"/>
      <xsd:attribute name="UpperIndex" type="xsd:NMTOKEN" use="required"/>
   </xsd:complexType>
</xsd:element>
<xsd:element name="Author" type="xsd:string"/>
<xsd:element name="Automatic">
   <xsd:complexType/>
</xsd:element>
<xsd:element name="BasicType">
   <xsd:complexType>
      <xsd:attribute name="Type" use="required">
         <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="STRING"/>
               <xsd:enumeration value="FLOAT"/>
               <xsd:enumeration value="INTEGER"/>
               <xsd:enumeration value="REFERENCE"/>
               <xsd:enumeration value="DATETIME"/>
               <xsd:enumeration value="BOOLEAN"/>
               <xsd:enumeration value="PERFORMER"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:attribute>
   </xsd:complexType>
```

```
    </xsd:element>
    <xsd:element name="BlockActivity">
       <xsd:complexType>
          <xsd:attribute name="BlockId" type="xsd:NMTOKEN" use="required"/>
       </xsd:complexType>
    </xsd:element>
    <xsd:element name="Codepage" type="xsd:string"/>
    <xsd:element name="Condition">
       <xsd:complexType mixed="true">
          <xsd:choice minOccurs="0" maxOccurs="unbounded">
             <xsd:element ref="xpdl:Xpression"/>
          </xsd:choice>
          <xsd:attribute name="Type">
             <xsd:simpleType>
                <xsd:restriction base="xsd:NMTOKEN">
                   <xsd:enumeration value="CONDITION"/>
                   <xsd:enumeration value="OTHERWISE"/>
                   <xsd:enumeration value="EXCEPTION"/>
                   <xsd:enumeration value="DEFAULTEXCEPTION"/>
                </xsd:restriction>
             </xsd:simpleType>
          </xsd:attribute>
       </xsd:complexType>
    </xsd:element>
    <xsd:element name="ConformanceClass">
       <xsd:complexType>
          <xsd:attribute name="GraphConformance">
             <xsd:simpleType>
                <xsd:restriction base="xsd:NMTOKEN">
                   <xsd:enumeration value="FULL_BLOCKED"/>
                   <xsd:enumeration value="LOOP_BLOCKED"/>
                   <xsd:enumeration value="NON_BLOCKED"/>
                </xsd:restriction>
             </xsd:simpleType>
          </xsd:attribute>
       </xsd:complexType>
    </xsd:element>
    <xsd:element name="Cost" type="xsd:string"/>
    <xsd:element name="CostUnit" type="xsd:string"/>
    <xsd:element name="Countrykey" type="xsd:string"/>
    <xsd:element name="Created" type="xsd:string"/>
    <xsd:element name="DataField">
       <xsd:complexType>
          <xsd:sequence>
             <xsd:element ref="xpdl:DataType"/>
             <xsd:element ref="xpdl:InitialValue" minOccurs="0"/>
             <xsd:element ref="xpdl:Length" minOccurs="0"/>
             <xsd:element ref="xpdl:Description" minOccurs="0"/>
             <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
          <xsd:attribute name="Name" type="xsd:string"/>
          <xsd:attribute name="IsArray" default="FALSE">
             <xsd:simpleType>
                <xsd:restriction base="xsd:NMTOKEN">
                   <xsd:enumeration value="TRUE"/>
                   <xsd:enumeration value="FALSE"/>
                </xsd:restriction>
             </xsd:simpleType>
          </xsd:attribute>
       </xsd:complexType>
    </xsd:element>
    <xsd:element name="DataFields">
```

```
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:DataField" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="DataType">
      <xsd:complexType>
         <xsd:group ref="xpdl:DataTypes"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:group name="DataTypes">
      <xsd:choice>
         <xsd:element ref="xpdl:BasicType"/>
         <xsd:element ref="xpdl:DeclaredType"/>
         <xsd:element ref="xpdl:SchemaType"/>
         <xsd:element ref="xpdl:ExternalReference"/>
         <xsd:element ref="xpdl:RecordType"/>
         <xsd:element ref="xpdl:UnionType"/>
         <xsd:element ref="xpdl:EnumerationType"/>
         <xsd:element ref="xpdl:ArrayType"/>
         <xsd:element ref="xpdl:ListType"/>
      </xsd:choice>
   </xsd:group>
   <xsd:element name="Deadline">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="DeadlineCondition" minOccurs="1" maxOccurs="1"/>
            <xsd:element name="ExceptionName" minOccurs="1" maxOccurs="1"/>
         </xsd:sequence>
         <xsd:attribute name="Execution">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="ASYNCHR"/>
                  <xsd:enumeration value="SYNCHR"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="DeclaredType">
      <xsd:complexType>
         <xsd:attribute name="Id" type="xsd:IDREF" use="required"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Description" type="xsd:string"/>
   <xsd:element name="Documentation" type="xsd:string"/>
   <xsd:element name="Duration" type="xsd:string"/>
   <xsd:element name="EnumerationType">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:EnumerationValue" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="EnumerationValue">
      <xsd:complexType>
         <xsd:attribute name="Name" type="xsd:NMTOKEN" use="required"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ExtendedAttribute">
      <xsd:complexType mixed="true">
         <xsd:choice minOccurs="0" maxOccurs="unbounded">
```

```
            <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:choice>
         <xsd:attribute name="Name" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="Value" type="xsd:string"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ExtendedAttributes">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:ExtendedAttribute" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ExternalPackage">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="href" type="xsd:string"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ExternalPackages">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:ExternalPackage" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ExternalReference">
      <xsd:complexType>
         <xsd:attribute name="xref" type="xsd:NMTOKEN" use="optional"/>
         <xsd:attribute name="location" type="xsd:anyURI" use="required"/>
         <xsd:attribute name="namespace" type="xsd:anyURI" use="optional"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="FinishMode">
      <xsd:complexType>
         <xsd:choice>
            <xsd:element ref="xpdl:Automatic"/>
            <xsd:element ref="xpdl:Manual"/>
         </xsd:choice>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="FormalParameter">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:DataType"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="Index" type="xsd:NMTOKEN"/>
         <xsd:attribute name="Mode" default="IN">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="IN"/>
                  <xsd:enumeration value="OUT"/>
                  <xsd:enumeration value="INOUT"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="FormalParameters">
```

```
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:FormalParameter" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Icon" type="xsd:string"/>
   <xsd:element name="Implementation">
      <xsd:complexType>
         <xsd:choice>
            <xsd:element ref="xpdl:No"/>
            <xsd:element ref="xpdl:Tool" maxOccurs="unbounded"/>
            <xsd:element ref="xpdl:SubFlow"/>
         </xsd:choice>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="InitialValue" type="xsd:string"/>
   <xsd:element name="Join">
      <xsd:complexType>
         <xsd:attribute name="Type">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="AND"/>
                  <xsd:enumeration value="XOR"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Length" type="xsd:string"/>
   <xsd:element name="Limit" type="xsd:string"/>
   <xsd:element name="ListType">
      <xsd:complexType>
         <xsd:group ref="xpdl:DataTypes"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Manual">
      <xsd:complexType/>
   </xsd:element>
   <xsd:element name="Member">
      <xsd:complexType>
         <xsd:group ref="xpdl:DataTypes"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="No">
      <xsd:complexType/>
   </xsd:element>
   <xsd:element name="Package">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:PackageHeader"/>
            <xsd:element ref="xpdl:RedefinableHeader" minOccurs="0"/>
            <xsd:element ref="xpdl:ConformanceClass" minOccurs="0"/>
            <xsd:element ref="xpdl:Script" minOccurs="0"/>
            <xsd:element ref="xpdl:ExternalPackages" minOccurs="0"/>
            <xsd:element ref="xpdl:TypeDeclarations" minOccurs="0"/>
            <xsd:element ref="xpdl:Participants" minOccurs="0"/>
            <xsd:element ref="xpdl:Applications" minOccurs="0"/>
            <xsd:element ref="xpdl:DataFields" minOccurs="0"/>
            <xsd:element ref="xpdl:WorkflowProcesses" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
```

```
        <xsd:attribute name="Name" type="xsd:string"/>
    </xsd:complexType>
</xsd:element>
<xsd:element name="PackageHeader">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:XPDLVersion"/>
            <xsd:element ref="xpdl:Vendor"/>
            <xsd:element ref="xpdl:Created"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:Documentation" minOccurs="0"/>
            <xsd:element ref="xpdl:PriorityUnit" minOccurs="0"/>
            <xsd:element ref="xpdl:CostUnit" minOccurs="0"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
<xsd:element name="Participant">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:ParticipantType"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:ExternalReference" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
        <xsd:attribute name="Name" type="xsd:string"/>
    </xsd:complexType>
</xsd:element>
<xsd:element name="ParticipantType">
    <xsd:complexType>
        <xsd:attribute name="Type" use="required">
            <xsd:simpleType>
                <xsd:restriction base="xsd:NMTOKEN">
                    <xsd:enumeration value="RESOURCE_SET"/>
                    <xsd:enumeration value="RESOURCE"/>
                    <xsd:enumeration value="ROLE"/>
                    <xsd:enumeration value="ORGANIZATIONAL_UNIT"/>
                    <xsd:enumeration value="HUMAN"/>
                    <xsd:enumeration value="SYSTEM"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:attribute>
    </xsd:complexType>
</xsd:element>
<xsd:element name="Participants">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:Participant" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
<xsd:element name="Performer" type="xsd:string"/>
<xsd:element name="Priority" type="xsd:string"/>
<xsd:element name="PriorityUnit" type="xsd:string"/>
<xsd:element name="ProcessHeader">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:Created" minOccurs="0"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:Priority" minOccurs="0"/>
            <xsd:element ref="xpdl:Limit" minOccurs="0"/>
            <xsd:element ref="xpdl:ValidFrom" minOccurs="0"/>
            <xsd:element ref="xpdl:ValidTo" minOccurs="0"/>
```

```
            <xsd:element ref="xpdl:TimeEstimation" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="DurationUnit">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="Y"/>
                  <xsd:enumeration value="M"/>
                  <xsd:enumeration value="D"/>
                  <xsd:enumeration value="h"/>
                  <xsd:enumeration value="m"/>
                  <xsd:enumeration value="s"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="RecordType">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Member" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="RedefinableHeader">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Author" minOccurs="0"/>
            <xsd:element ref="xpdl:Version" minOccurs="0"/>
            <xsd:element ref="xpdl:Codepage" minOccurs="0"/>
            <xsd:element ref="xpdl:Countrykey" minOccurs="0"/>
            <xsd:element ref="xpdl:Responsibles" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="PublicationStatus">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="UNDER_REVISION"/>
                  <xsd:enumeration value="RELEASED"/>
                  <xsd:enumeration value="UNDER_TEST"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Responsible" type="xsd:string"/>
   <xsd:element name="Responsibles">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Responsible" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Route">
      <xsd:complexType/>
   </xsd:element>
   <xsd:element name="SchemaType">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Script">
      <xsd:complexType>
```

```
      <xsd:attribute name="Type" type="xsd:string" use="required"/>
      <xsd:attribute name="Version" type="xsd:string" use="optional"/>
      <xsd:attribute name="Grammar" type="xsd:anyURI" use="optional"/>
   </xsd:complexType>
</xsd:element>
<xsd:element name="SimulationInformation">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref="xpdl:Cost"/>
         <xsd:element ref="xpdl:TimeEstimation"/>
      </xsd:sequence>
      <xsd:attribute name="Instantiation">
         <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="ONCE"/>
               <xsd:enumeration value="MULTIPLE"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:attribute>
   </xsd:complexType>
</xsd:element>
<xsd:element name="Split">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref="xpdl:TransitionRefs" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="Type">
         <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="AND"/>
               <xsd:enumeration value="XOR"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:attribute>
   </xsd:complexType>
</xsd:element>
<xsd:element name="StartMode">
   <xsd:complexType>
      <xsd:choice>
         <xsd:element ref="xpdl:Automatic"/>
         <xsd:element ref="xpdl:Manual"/>
      </xsd:choice>
   </xsd:complexType>
</xsd:element>
<xsd:element name="SubFlow">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref="xpdl:ActualParameters" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="Id" type="xsd:string" use="required"/>
      <xsd:attribute name="Execution">
         <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="ASYNCHR"/>
               <xsd:enumeration value="SYNCHR"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:attribute>
   </xsd:complexType>
</xsd:element>
<xsd:element name="TimeEstimation">
   <xsd:complexType>
      <xsd:sequence>
```

```
            <xsd:element ref="xpdl:WaitingTime" minOccurs="0"/>
            <xsd:element ref="xpdl:WorkingTime" minOccurs="0"/>
            <xsd:element ref="xpdl:Duration" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Tool">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:ActualParameters" minOccurs="0"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="Type">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="APPLICATION"/>
                  <xsd:enumeration value="PROCEDURE"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Transition">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Condition" minOccurs="0"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="From" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="To" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="Name" type="xsd:string"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="TransitionRef">
      <xsd:complexType>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="TransitionRefs">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:TransitionRef" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="TransitionRestriction">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Join" minOccurs="0"/>
            <xsd:element ref="xpdl:Split" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="TransitionRestrictions">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:TransitionRestriction" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
```

```
    </xsd:element>
    <xsd:element name="Transitions">
       <xsd:complexType>
          <xsd:sequence>
             <xsd:element ref="xpdl:Transition" minOccurs="0" maxOccurs="unbounded"/>
          </xsd:sequence>
       </xsd:complexType>
    </xsd:element>
    <xsd:element name="TypeDeclaration">
       <xsd:complexType>
          <xsd:sequence>
             <xsd:group ref="xpdl:DataTypes"/>
             <xsd:element ref="xpdl:Description" minOccurs="0"/>
             <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required"/>
          <xsd:attribute name="Name" type="xsd:string"/>
       </xsd:complexType>
    </xsd:element>
    <xsd:element name="TypeDeclarations">
       <xsd:complexType>
          <xsd:sequence>
             <xsd:element ref="xpdl:TypeDeclaration" minOccurs="0" maxOccurs="unbounded"/>
          </xsd:sequence>
       </xsd:complexType>
    </xsd:element>
    <xsd:element name="UnionType">
       <xsd:complexType>
          <xsd:sequence>
             <xsd:element ref="xpdl:Member" maxOccurs="unbounded"/>
          </xsd:sequence>
       </xsd:complexType>
    </xsd:element>
    <xsd:element name="ValidFrom" type="xsd:string"/>
    <xsd:element name="ValidTo" type="xsd:string"/>
    <xsd:element name="Vendor" type="xsd:string"/>
    <xsd:element name="Version" type="xsd:string"/>
    <xsd:element name="WaitingTime" type="xsd:string"/>
    <xsd:element name="WorkflowProcess">
       <xsd:complexType>
          <xsd:sequence>
             <xsd:element ref="xpdl:ProcessHeader"/>
             <xsd:element ref="xpdl:RedefinableHeader" minOccurs="0"/>
             <xsd:element ref="xpdl:FormalParameters" minOccurs="0"/>
             <xsd:element ref="xpdl:DataFields" minOccurs="0"/>
             <xsd:element ref="xpdl:Participants" minOccurs="0"/>
             <xsd:element ref="xpdl:Applications" minOccurs="0"/>
             <xsd:element ref="xpdl:ActivitySets" minOccurs="0"/>
             <xsd:element ref="xpdl:Activities" minOccurs="0"/>
             <xsd:element ref="xpdl:Transitions" minOccurs="0"/>
             <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
          <xsd:attribute name="Name" type="xsd:string"/>
          <xsd:attribute name="AccessLevel">
             <xsd:simpleType>
                <xsd:restriction base="xsd:NMTOKEN">
                   <xsd:enumeration value="PUBLIC"/>
                   <xsd:enumeration value="PRIVATE"/>
                </xsd:restriction>
             </xsd:simpleType>
          </xsd:attribute>
       </xsd:complexType>
```

```
    </xsd:element>
    <xsd:element name="WorkflowProcesses">
       <xsd:complexType>
          <xsd:sequence>
             <xsd:element ref="xpdl:WorkflowProcess" minOccurs="0" maxOccurs="unbounded"/>
          </xsd:sequence>
       </xsd:complexType>
    </xsd:element>
    <xsd:element name="WorkingTime" type="xsd:string"/>
    <xsd:element name="XPDLVersion" type="xsd:string"/>
    <xsd:element name="Xpression">
       <xsd:complexType mixed="true">
          <xsd:choice minOccurs="0" maxOccurs="unbounded">
             <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
          </xsd:choice>
       </xsd:complexType>
    </xsd:element>
</xsd:schema>
```

1.2　遷移条件式スキーマ仕様

本ワークフロー運用基盤で使用する遷移条件定義用の XML スキーマ定義を示す。このスキーマはすべてのタグを使用する。

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace=http://www.argo21.co.jp/argonoautica/workflow
 xmlns:awf=http://www.argo21.co.jp/argonoautica/workflow
 xmlns:xsd=http://www.w3.org/2001/XMLSchema
 elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xsd:element name="ConditionalExpression">
      <xsd:complexType>
         <xsd:group ref="awf:ConditionalExpressionGroup" minOccurs="1" maxOccurs="1"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:group name="ConditionalExpressionGroup">
      <xsd:choice>
         <xsd:element ref="awf:UnaryExpression"/>
         <xsd:element ref="awf:LogicalExpression"/>
         <xsd:element ref="awf:RelationalExpression"/>
      </xsd:choice>
   </xsd:group>
   <xsd:element name="UnaryExpression">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:group ref="awf:ConditionalExpressionGroup" minOccurs="1" maxOccurs="1"/>
         </xsd:sequence>
         <xsd:attribute name="Type" use="required">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="NOT"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="LogicalExpression">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:group ref="awf:ConditionalExpressionGroup" minOccurs="2" maxOccurs="2"/>
```

```
         </xsd:sequence>
         <xsd:attribute name="Type" use="required">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="OR"/>
                  <xsd:enumeration value="AND"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="RelationalExpression">
      <xsd:complexType>
         <xsd:attribute name="DataFieldRef" type="xsd:string" use="required"/>
         <xsd:attribute name="ComparedValue" type="xsd:string" use="required"/>
         <xsd:attribute name="Type" use="required">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="EQUAL"/>
                  <xsd:enumeration value="NOT_EQUAL"/>
                  <xsd:enumeration value="LESS_THAN"/>
                  <xsd:enumeration value="LESS_EQUAL"/>
                  <xsd:enumeration value="GREATER_THAN"/>
                  <xsd:enumeration value="GREATER_EQUAL"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
</xsd:schema>
```

2　DB テーブル仕様

2.1　テーブルー覧

No	テーブル名	テーブル ID	M/T	備考
1	キーシーケンス管理	KEYSEQ	T	ユニークキー取得用
2	ワークフロー定義管理	DEFINITION	M	
3	組織	ORGANIZATION	M	
4	ユーザ	USER	M	
5	役割	ROLE	M	
6	組織ユーザ関連	ORGUSERMAP	M	
7	組織役割関連	ORGROLEMAP	M	
8	ユーザ役割関連	USERROLEMAP	M	
10	プロセスインスタンス	PROCESS	T	
11	プロセス変数インスタンス	VARIABLE	T	
12	アクティビティインスタンス	ACTIVITY	T	
13	作業項目	WORKITEM	T	
14	プロセス完了待ちアクティビティ	WAITPROCACT	T	
15	並行処理ブロック	CONCURBLOCK	T	
16	アプリケーション実行	APPEXEC	T	
17	プロセス状態変更履歴	PROCESSHIST	T	
18	プロセス変数変更履歴	VARIABLEHIST	T	
19	アクティビティ状態変更履歴	ACTIVITYHIST	T	
20	起動先プロセス	ETAPROCESS	T	エンジン間連携用
21	起動元アクティビティ	ETAACTIVITY	T	エンジン間連携用

2.2　テーブル詳細

2.2.1　キーシーケンス管理（KEYSEQ）

項目 ID	項目名	型	サイズ	P	N	初期値	説明
NAME	キーシーケンス名	文字列	32	○	○		
SEQ	シーケンス	数値	19		○	0	取得されるたびに+1

2.2.2　ワークフロー定義管理（DEFINITION）

項目 ID	項目名	型	サイズ	P	N	初期値	説明
NAME	定義名	文字列	128	○	○		
PATH	定義相対パス	文字列	128		○		

PROC_DEF_ID	プロセス定義 ID	文字列	64		○		
REGISTER_DATE	登録日	日付			○		
DEF_STATE	定義状態	数値	1		○		enable/disable

2.2.3 組織(ORGANIZATION)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
ORG_ID	組織 ID	文字列	64	○	○		
ORG_NAME	組織名	文字列	128		○		
RESPONSIBLE_ID	責任者 ID	文字列	64				
HIGHER_ORG_ID	上位組織 ID	文字列	64				

2.2.4 ユーザ(USER)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
USER_ID	ユーザ ID	文字列	64	○	○		
FIRST_NAME	名	文字列	64		○		
LAST_NAME	姓	文字列	64		○		
PASSWORD	パスワード	文字列	64		○		
MAIL_ADDRESS	メールアドレス	文字列	64		○		必須

2.2.5 役割(ROLE)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
ROLE_ID	役割 ID	文字列	64	○	○		
ROLE_NAME	役割名	文字列	128		○		

2.2.6 組織ユーザ関連(ORGUSERMAP)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
ORG_ID	組織 ID	文字列	64	○	○		
USER_ID	ユーザ ID	文字列	64	○	○		

2.2.7 組織役割関連(ORGROLEMAP)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
ROLE_ID	役割 ID	文字列	64	○	○		
ORG_ID	組織 ID	文字列	64	○	○		

2.2.8 ユーザ役割関連(USERROLEMAP)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
ROLE_ID	役割 ID	文字列	64	○	○		
USER_ID	ユーザ ID	文字列	64	○	○		

2.2.9 プロセスインスタンス(PROCESS)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
PROC_ID	プロセスインスタ	文字列	64	○	○		

	ンス ID						
PACKAGE_ID	パッケージ ID	文字列	64		○		
PROC_DEF_ID	プロセス定義 ID	文字列	64		○		
START_DATE	開始日時	日時			○		
STARTER_ID	開始者	文字列	64		○		開始者のセッション ID。実行者がエンジンの場合は、`__SYSTEM__’を設定する。
DURATION_LIMIT	実行期限	日時			○		0 は無期限
PROC_STATE	状態	数値	1		○	not_started	
UPDATE_DATE	最新状態更新日時	日時					

2.2.10 プロセス変数インスタンス（VARIABLE）

項目 ID	項目名	型	サイズ	P	N	初期値	説明
VAR_ID	プロセス変数インスタンス ID	文字列	64	○	○		
PROC_ID	プロセスインスタンス ID	文字列	64		○		
VAR_DEF_ID	プロセス変数定義 ID	文字列	64		○		
TYPE	変数型	数値	1		○		文字列値 数値 真理値
VALUE	変数値	文字列	512		○		値の文字列表現
UPDATE_DATE	最新状態更新日時	日時					

2.2.11 アクティビティインスタンス(ACTIVITY)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
PROC_ID	プロセスインスタンス ID	文字列	64	○	○		
ACT_ID	アクティビティインスタンス ID	文字列	64	○	○		
ACT_DEF_ID	アクティビティ定義 ID	文字列	64		○		
START_DATE	開始日時	日時			○		
STARTER_ID	開始者	文字列	64				

DURATION_LIMIT	実行期限	日時			○		0 は無期限
BLOCK_ID	所属する並行処理ブロック ID	文字列	64				ブロックに所属していない場合は、NULL
ACT_STATE	状態	数値	1		○	not_started	
ACTOR_ID	実行者 ID	文字列	64				実行者のセッション ID。実行者がエンジンの場合は、`__SYSTEM__’を設定する。
UPDATE_DATE	最新状態更新日時	日時					

2.2.12 作業項目(WORKITEM)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
WORK_ID	作業項目 ID	文字列	64	○	○		
PROC_ID	プロセスインスタンス ID	文字列	64		○		
ACT_ID	アクティビティインスタンス ID	文字列	64		○		
PARTICIPANT_TYPE	ワークフロー参加者型	数値	1				ユーザ 組織単位 ロール システム
PARTICIPANT_NAME	ワークフロー参加者名称	文字列	128				

2.2.13 プロセス完了待ちアクティビティ(WAITPROCACT)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
PROC_ID	プロセスインスタンス ID	文字列	64		○		
ACT_ID	完了待ちアクティビティインスタンス ID	文字列	64		○		
SUBPROC_ID	サブプロセスインスタンス ID	文字列	64		○		

2.2.14 並行処理ブロック(CONCURBLOCK)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
BLOCK_ID	ブロック ID	文字列	64		○		

UPPER_BLOCK_ID	上位のブロック ID	文字列	64				最上位のブロックの場合は、NULL
PROC_ID	プロセスインスタンス ID	文字列	64		○		
ACT_ID	ブロック開始アクティビティインスタンス ID	文字列	64		○		

2.2.15 アプリケーション実行(APPEXEC)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
APP_NAME	アプリケーション名	文字列	64	○	○		
TOOL_AGENT	アプリケーションを起動したツールエージェントのクラス名	文字列	64	○	○		
EXEC_ID	ツールエージェントが発行した実行 ID	数値	19	○	○		
START_DATE	開始日時	日時					
STARTER_ID	実行者	文字列	64				
END_DATE	終了日時	日時					
APP_STATE	実行状態	数値	1			not_started	

2.2.16 プロセス状態変更履歴(PROCESSHIST)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
PROC_ID	プロセスインスタンス ID	文字列	64		○		
PACKAGE_ID	パッケージ ID	文字列	64		○		
PROC_DEF_ID	プロセス定義 ID	文字列	64		○		
PREV_ STATE	更新前の状態	数値	1				
NEW_STATE	更新後の状態	数値	1				
UPDATE_DATE	更新日時	日時					
REASON	更新事由	文字列	128				

2.2.17 プロセス変数変更履歴（VARIABLEHIST）

項目 ID	項目名	型	サイズ	P	N	初期値	説明
VAR_ID	プロセス変数イン	文字列	64		○		

	スタンス ID						
PROC_ID	プロセスインスタンス ID	文字列	64		○		
VAR_DEF_ID	プロセス変数定義 ID	文字列	64		○		
TYPE	変数型	数値	1		○		文字列値 数値 真理値
PREV_ VALUE	更新前の変数値	文字列	512				
NEW_VALUE	更新後の変数値	文字列	512		○		
UPDATE_DATE	更新日時	日時			○		
REASON	更新事由	文字列	128				
ACT_ID	変更アクティビティ ID	文字列	64				

2.2.18 アクティビティ状態変更履歴(ACTIVITYHIST)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
PROC_ID	プロセスインスタンス ID	文字列	64		○		
ACT_ID	アクティビティインスタンス ID	文字列	64		○		
ACT_DEF_ID	アクティビティ定義 ID	文字列	64		○		
ACTOR_ID	実行者 ID	文字列	64				実行者のセッション ID。実行者がエンジンの場合は、`__SYSTEM__’を設定する。
PREV_ STATE	更新前の状態	数値	1				
NEW_STATE	更新後の状態	数値	1		○		
UPDATE_DATE	更新日時	日時			○		
REASON	更新事由	文字列	128				

2.2.19 起動先プロセス(ETAPROCESS)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
ENGINE_ID	起動先エンジン ID	文字列	64	○	○		
PROCESS_ID	起動先プロセス ID	文字列	64	○	○		

ROOT_PROC_ID	起動元プロセス ID	文字列	64		○		
ROOT_ACT_ID	起動元アクティビティ ID	文字列	64		○		

2.2.20 起動元アクティビティ(ETAACTIVITY)

項目 ID	項目名	型	サイズ	P	N	初期値	説明
PROCESS_ID	起動先プロセス ID	文字列	64	○	○		
ENGINE_ID	起動元エンジン ID	文字列	64		○		
ROOT_PROC_ID	起動元プロセス ID	文字列	64		○		
ROOT_ACT_ID	起動元アクティビティ ID	文字列	64		○		

IV 資料編

1　商用ワークフローシステム運用基盤一覧

販売元	製品名	価格	WfMC 対応				
			IF1	IF2	IF3	IF4	IF5
NEC ソフト	ActiveFlow	サーバ 1,200,000 ツール 300,000 10 クライアント 100,000	○				
ファイルネット	PANAGON	要問合わせ				○	
日立製作所	Groupmax	1,890,000				○	
京セラコミュニケーションシステム セイコーエプソン	GreenOffice	6,000,000/1CPU～	不明	不明	不明	不明	不明
日本アイ・ビー・エム	FormWave	4,097,000～	○	○		○	○
日本ティブコソフトウェア	ActiveEnterprise	要問合わせ	○	○	○	○	
日本ユニシス	Staffware	要問合わせ	○	○	○	○	○

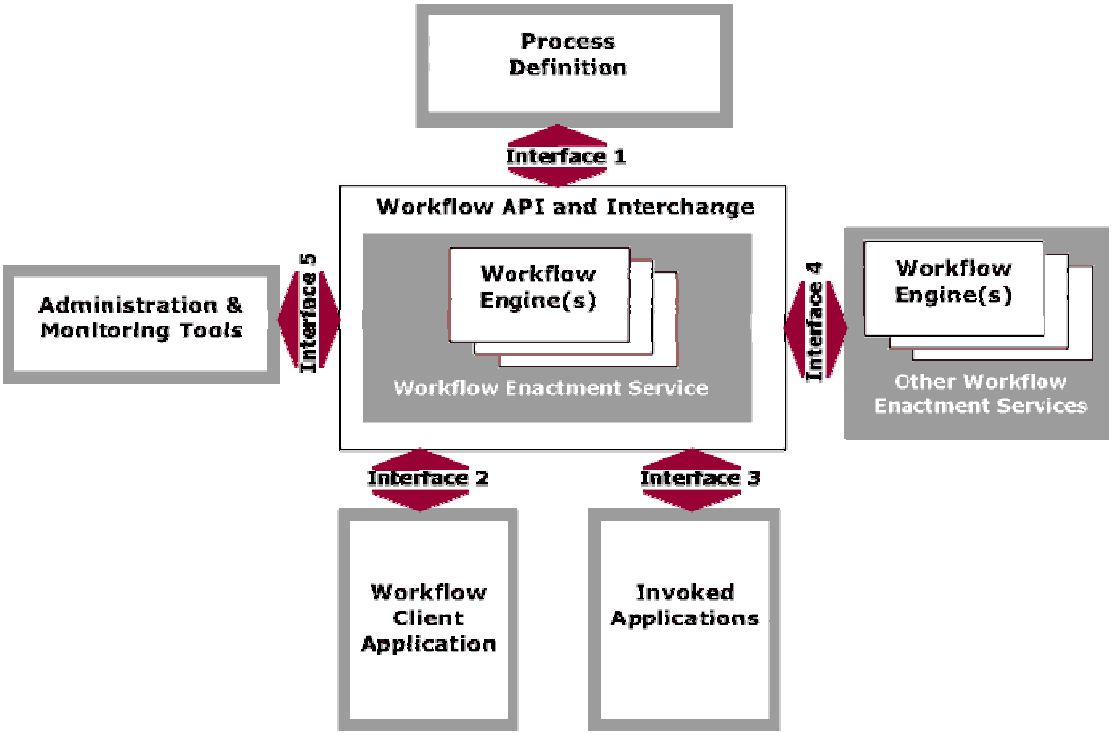
・代表的な商用製品 26 種から WfMC 対応を行っているものを抜粋（2004 年 10 月時調査）

2　主な OSS ワークフローシステム運用基盤ソフトウェアの比較

		本システム	Enhydra Shark	ObjectWeb Bonita	OpenSymphony Workflow	OpenWFE
ライセンス形態		LGPL	LGPL	LGPL	Apache	BSD
EJB コンテナへの対応		○	○	○	○	○
Servlet コンテナへの対応		○	×	×	×	○
別アプリケーションへの組み込み		○	×	×	×	×
ワークフロー定義ツール		○	○	○	○	○
ワークフロー運用管理ツール		○	○	○	×	○
Web サービスへの対応		○	○	×	○	×
WfMC	インタフェース-1 定義仕様	準拠 XPDL	準拠 XPDL	×	×	×
	インタフェース-2 & 3 ワークフローAPI	準拠	一部 準拠	×	×	×
	インタフェース-4 エンジン連携 API	準拠	予定	×	×	×
	インタフェース-5 監査データ仕様	準拠	一部 準拠	×	×	×

・Google による検索結果上位のものを選定（2004 年 10 月時調査）

3　WfMC のワークフロー参照モデルと策定状況



(1) インタフェース 1

ワークフロー定義に必要なデータ形式に関する仕様。策定時には、WPDL(Workflow Process Definition Language)という独自形式であったが、現在は XML を使用して定義しなおした XPDL(XML Process Definition Language)仕様が公開されている。

(2) インタフェース 2 & 3

ワークフローエンジンが持つべき外部プログラミングインタフェースに関する仕様。もともとは二つのインタフェース仕様であったが、現在は一つに統合されている。基本的に C 言語を想定した構成となっているため、他言語による実装では、機能レベルの対応となり、対応表を用意することになる。

(3) インタフェース 4

ワークフローエンジン間で連携を行うための外部プログラミングインタフェースに関する仕様。現在の時点でドラフト版である。2 & 3 と同様に C 言語を想定した構成となっているため、他言語による実装では、機能レベルの対応となり、対応表を用意することになる。

(4) インタフェース 5

ワークフローエンジン運用時に監査すべきデータに関する仕様。これらには、ログ

等により記録すべき情報が記載されている。

4 WfMC ワークフローAPI 対応表

以下に WfMC インタフェースに対する本ワークフロー運用基盤における対応を示す。WfMC の各インタフェースは、主に C 言語による実装を想定しており、他言語においては言語仕様に合わせた機能レベルのマッピングを行う必要がある。本ワークフロー運用基盤では、実装言語として Java を採用したので、Java の言語仕様に基づいたマッピングを行った。

4.1 インタフェース１

インタフェース１については、WfMC が提供する XPDL(XML Process Definition Language)仕様に準拠することで対応とした。スキーマ構造については、「III.入出力データ編」を参照のこと。

4.2 インタフェース２＆３

4.2.1 基本データ型

仕様書項番	機能名	仕様書記述の API（C 言語） 本ワークフロー運用基盤の API（Java）
	符号付 8bit 整数	typedef char WMTInt8; 本ワークフロー運用基盤では、すべて int 型として数値を扱うこととする。
	符号付 16bit 整数	typedef short WMTInt16; 本ワークフロー運用基盤では、すべて int 型として数値を扱うこととする。
	符号付 32bit 整数	typedef long WMTInt32; int
	符号なし 8bit 整数	typedef unsigned char WMTUInt8; Java には符号なし整数はない。本ワークフロー運用基盤では、すべて int 型として数値を扱うこととする。
	符号なし 16bit 整数	typedef unsigned short WMTUInt16; Java には符号なし整数はない。本ワークフロー運用基盤では、すべて int 型として数値を扱うこととする。
	符号なし 32bit 整数	typedef unsigned long WMTUInt32; Java には符号なし整数はない。本ワークフロー運用基盤では、すべて int 型として数値を扱うこととする。
	文字	typedef WMTInt8 WMTText; Java ではすべての文字を 16bit Unicode として扱う。そのため、適用局面では 16bit char 型として扱う。
	文字列	typedef WMTText *WMTPText; Java はすべての文字列を String クラスのオブジェクトとして扱う。
	符号付 8bit 整数へのポインタ	typedef WMTInt8 *WMTPInt8; Java には明示的なポインタはない。必要に応じて、Integer クラスを使用する。
	符号付 16bit 整数へのポインタ	typedef WMTInt16 *WMTPInt16; Java には明示的なポインタはない。必要に応じて、Integer クラスを使用する。
	符号付 32bit 整数へのポインタ	Typedef WMTInt32 *WMTPInt32; Java には明示的なポインタはない。必要に応じて、Integer クラスを使用する。
	真理値	Typedef WMTInt8 WMTBoolean;

		boolean 型とする。
	符号なし 8bit 整数へのポインタ	typedef WMTUInt8 *WMTPointer;
		Java には明示的なポインタはない。必要に応じて、Object 型、もしくは任意のサブクラスを定義し使用する。
	プライベートなデータへのポインタ	typedef WMTText *WMTPPrivate;
		Java には明示的なポインタはない。必要に応じて、Object 型、もしくは任意のサブクラスを定義し使用する。
	NULL を表す値	#define WMNULL ((WMTPointer)0)
		予約語 null とする。
	真理値の偽を表す値	#define WMFalse 0
		予約語 false とする。
	真理値の真を表す値	#define WMTrue (!WMFalse)
		予約語 true とする。

4.2.2 その他のデータ型

仕様書 項番	機能名	仕様書記述の API（C 言語） 本ワークフロー運用基盤の API（Java）
	名前の最大長を表す定数	#define NAME_STRING_SIZE 64 WorkflowConstants インタフェースを提供する。 static final int NAME_STRING_SIZE = 64;
	ユーザ ID	WMTText user_identification[NAME_STRING_SIZE]; String クラスとする。
	ID の最大長を表す定数	#define UNIQUE_ID_SIZE 64 WorkflowConstants インタフェースを提供する。 static final int UNIQUE_ID_SIZE = 64; また、ID は一意な値を持つ String 型として扱う。

	エラーコード	typedef struct
		{
		WMTInt16 main_code;
		WMTInt16 sub_code;
		} WMTErrRetType;
		Java には構造体はない。
		本ワークフロー運用基盤では、エラーコードはすべて例外にマッピングする。すべての例外の基底クラスとして、WorkflowException を提供する。
		public class WorkflowException extends Exception {
		public int getMainCode();
		public int getSubCode();
		}

	接続情報	typedef struct
		{
		WMTText user_identification[NAME_STRING_SIZE];
		WMTText password[NAME_STRING_SIZE];
		WMTText engine_name[NAME_STRING_SIZE];
		WMTText scope[NAME_STRING_SIZE];
		} WMTConnectInfo;
		typedef WMTConnectInfo *WMTPConnectInfo;
		Java には構造体はない。
		以下のクラス定義により、代替する。
		public class ConnectionInfo {
		String userIdentification;
		String password;
		String engineName;
		String scope;
		}

	セッション情報	typedef struct { WMTUInt32 session_id; WMTPPrivate pprivate; } WMTSessionHandle; typedef WMTSessionHandle *WMTPSessionHandle;

		Java には構造体はない。 Session クラス定義により、代替する。
	フィルター	typedef struct { WMTInt32 filter_type; WMTInt32 filter_length; WMTText attribute_name[NAME_STRING_SIZE]; WMTUInt32 comparison; WMTPText filter_string; } WMTFilter; typedef WMTFilter *WMTPFilter;

		Java には構造体はない。
		以下のクラス定義により、代替する。
		public class Filter {
		int filterType;
		String attributeName;
		int comparison;
		String filterString;
		}

	一覧に対する問い合わせハンドル	typedef struct { WMTUInt32 query_handle; }WMTQueryHandle; typedef WMTQueryHandle *WMTPQueryHandle;
		Java には構造体はない。 型に合わせた問い合わせを実装したクラスにより配列を操作することによって代替する。
	ワークフロー参加者	typedef struct { WMTText wf_participant[NAME_STRING_SIZE]; } WMTWflParticipant; typedef WMTWflParticipant *WMTPWflParticipant;
		Java には構造体はない。 WfResource クラス定義により、代替する。
	プロセス定義 ID	typedef struct { WMTText proc_def_id[UNIQUE_ID_SIZE]; } WMTProcDefID; typedef WMTProcDefID *WMTPProcDefID;
		Java には構造体はない。 ID は一意な値を持つ String 型として扱う。
	アクティビティ定義 ID	typedef struct { WMTText activity_id[NAME_STRING_SIZE]; }WMTActivityID; typedef WMTActivityID *WMTPActivityID;
		Java には構造体はない。 ID は一意な値を持つ String 型として扱う。
	プロセス定義状態	typedef struct { WMTText proc_def_state[NAME_STRING_SIZE]; } WMTProcDefState; typedef WMTProcDefState *WMTPProcDefState;

		Java には構造体はない。
		以下のクラス定義により、代替する。
		public class ProcessDefinitionState
		implements ObjectState {
		}

	プロセス定義	typedef struct { WMTText process_name[NAME_STRING_SIZE]; WMTProcDefID proc_def_id; WMTProcDefState state; } WMTProcDef; typedef WMTProcDef *WMTPProcDef;

		Java には構造体はない。
		以下のクラス定義により、代替する。
		public class ProcessDefinition {
		String name;
		String id;
		String state;
		}

	プロセスインスタンス ID	typedef struct { WMTText proc_inst_id[UNIQUE_ID_SIZE]; }WMTProcInstID;
		Java には構造体はない。 ID は一意な値を持つ String 型として扱う。
	プロセスインスタンス状態	typedef struct { WMTText proc_inst_state[NAME_STRING_SIZE]; } WMTProcInstState; typedef WMTProcInstState *WMTPProcInstState;

		Java には構造体はない。
		以下のクラス定義により、代替する。
		public class ProcessState implements ObjectState {
		}

	プロセスインスタンス	typedef struct { WMTText process_name[NAME_STRING_SIZE]; WMTProcInstID proc_inst_id; WMTProcDefID proc_def_id; WMTProcInstState state; WMTInt32 priority; WMTWflParticipant proc_participants[20]; } WMTProcInst; typedef WMTProcInst *WMTPProcInst;
		Java には構造体はない。 以下の WfProcess クラス定義により、代替する。
	アクティビティインスタンス ID	typedef struct { WMTText activity_inst_id[UNIQUE_ID_SIZE]; }WMTActivityInstID; typedef WMTActivityInstID *WMTPActivityInstID;
		Java には構造体はない。 ID は一意な値を持つ String 型として扱う。
	アクティビティインスタンス状態	typedef struct { WMTText activity_inst_state[NAME_STRING_SIZE]; } WMTActivityInstState; typedef WMTActivityInstState *WMTPActivityInstState;

		Java には構造体はない。
		以下のクラス定義により、代替する。
		public class ActivityState implements ObjectState {
		}

	アクティビティインスタンス	typedef struct { WMTText activity_name[NAME_STRING_SIZE]; WMTActivityInstID activity_inst_id; WMTProcInstID proc_inst_id; WMTActivityInstState state; WMTInt32 priority; WMTWflParticipant activity_participants[10]; } WMTActivityInst; typedef WMTActivityInst *WMTPActivityInst;
		Java には構造体はない。 WfActivity クラス定義により、代替する。
	作業項目 ID	typedef struct { WMTText work_item_id[UNIQUE_ID_SIZE]; }WMTWorkItemID; typedef WMTWorkItemID *WMTPWorkItemID;
		Java には構造体はない。 ID は一意な値を持つ String 型として扱う。
	作業項目	typedef struct { WMTText workitem_name[NAME_STRING_SIZE]; WMTWorkItemID workitem_id; WMTActivityInstID activity_inst_id; WMTProcInstID proc_inst_id; WMTInt32 priority; WMTWflParticipant participant; } WMTWorkItem; typedef WMTWorkItem *WMTPWorkItem;
		Java には構造体はない。 WorkItem クラス定義により、代替する。
	属性	typedef struct { WMTText attribute_name[NAME_STRING_SIZE]; WMTInt32 attribute_type; WMTInt32 attribute_length; WMTPText pattribute_value; } WMTAttribute; typedef WMTAttribute *WMTPAttribute;

		Java には構造体はない。
		以下のクラス定義により、代替する。
		public class Attribute {
		String name;
		int type;
		String value;
		}

	属性一覧	typedef struct { WMTInt32 attribute_number; WMTPAttribute pattribute; WMTNextAttr *WMTAttributeList } WMTAttributeList; typedef WMTAttributeList *WMTPAttributeList;
		不要なので使用しない。

4.2.3 属性

仕様書 項番	機能名	仕様書記述の API（C 言語）
		本ワークフロー運用基盤の API（Java）
	属性名	WMTText attribute_name[NAME_STRING_SIZE];
		String 型とする。
	属性型	WMTInt32 attribute_type;

		int 型とし、以下の値を WMWorkflowConstants インタフェースにて提供する。
		static final int ATTR_STRING_TYPE = 1;
		static final int ATTR_INTEGER_TYPE = 2;
		static final int ATTR_BOOLEAN_TYPE = 3;

	属性値の長さ	WMTInt32 attribute_length;
		不要なので使用しない。
	属性値	WMTPText pattribute_value;
		String 型とする。
	属性名	typedef WMTText WMTAttrName[NAME_STRING_SIZE]; typedef WMTAttrName *WMTPAttrName;
		不要なので使用しない。

4.2.4 エラーコード

仕様書 項番	機能名	仕様書記述の API（C 言語）
		本ワークフロー運用基盤の API（Java）
	ワークフローAPI エラー情報戻り値	WM_SUCCESS WM_CONNECT_FAILED WM_INVALID_PROCESS_DEFINITION WM_INVALID_ACTIVITY_NAME WM_INVALID_PROCESS_INSTANCE WM_INVALID_ACTIVITY_INSTANCE WM_INVALID_WORKITEM WM_INVALID_ATTRIBUTE WM_ATTRIBUTE_ASSIGNMENT_FAILED WM_INVALID_STATE WM_TRANSITION_NOT_ALLOWED WM_INVALID_SESSION_HANDLE WM_INVALID_QUERY_HANDLE WM_INVALID_SOURCE_USER WM_INVALID_TARGET_USER WM_INVALID_FILTER WM_LOCKED WM_NOT_LOCKED WM_NO_MORE_DATA WM_INSUFFICIENT_BUFFER_SIZE WM_APPLICATION_BUSY WM_INVALID_APPLICATION WM_INVALID_WORK_ITEM WM_APPLICATION_NOT_STARTED WM_APPLICATION_NOT_DEFINED WM_APPLICATION_NOT STOPPED

		WM_SUCCESS 以外を WorkflowException を基底クラスとする例外として実現する。 ConnectionFailedException InvalidProcessDefinitionException InvalidActivityNameException InvalidProcessInstanceException InvalidActivityInstanceException InvalidWorkItemException InvalidAttributeException AttributeAssignmentFailedException InvalidStateException TransitionNotAllowedException InvalidSessionException InvalidQueryException InvalidSourceUserException InvalidTargetUserException InvalidFilterException LockedException NotLockedException NoMoreDataException InsufficientBufferSizeException ApplicationBusyException InvalidApplicationException ApplicationNotStartedException ApplicationNotDefinedException ApplicationNotStoppedException

4.2.5 ワークフローAPI

(1) 接続機能

仕様書 項番	機能名	仕様書記述の API（C 言語） 本ワークフロー運用基盤の API（Java）
7.2.2	ワークフローエンジンへの接続	WMTErrRetType WMConnect（ in WMTPConnectInfo pconnect_info, out WMTPSessionHandle psession_handle）
		WorkflowEngineHandler インタフェースにおいて以下のメソッドを提供する。 Session connect(ConnectionInfo ci) throws WorkflowException
7.2.2	ワークフローエンジンとの切断	WMTErrRetType WMDisconnect（ in WMTPSessionHandle psession_handle）
		WorkflowEngineHandler インタフェースにおいて以下のメソッドを提供する。 void disconnect(Session session) throws WorkflowException

(2) プロセス制御機能

仕様書 項番	機能名	仕様書記述の API（C 言語） 本ワークフロー運用基盤の API（Java）
5.4.2	プロセス定義一覧のオープン	WMTErrRetType WMOpenProcessDefinitionsList（ in WMTPSessionHandle psession_handle, in WMTPFilter pproc_def_filter, in WMTBoolean count_flag, out WMTPQueryHandle pquery_handle, out WMTPInt32 pcount）
		クラスにおいて以下のメソッドを提供する。 ProcessDefinition[] getProcessDefinitions(Session session, Filter filter) throws WorkflowException

	プロセス定義のフェッチ	WMTErrRetType WMFetchProcessDefinition (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle, out WMTPProcDef pproc_def_buf_ptr)
		不要なので使用しない。
	プロセス定義一覧のクローズ	WMTErrRetType WMCloseProcessDefinitionsList(in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle)
		不要なので使用しない。
5.4.2	プロセス定義状態一覧のオープン	WMTErrRetType WMOpenProcessDefinitionStatesList (in WMTPSessionHandle psession_handle, in WMTPProcDefID pproc_def_id, in WMTPFilter pproc_def_state_filter, in WMTBoolean count_flag, out WMTPQueryHandle pquery_handle, out WMTUInt32 pcount)
		クラスにおいて以下のメソッドを提供する。 ProcessDefinitionState[] getProcessDefinitionStates (Session session, String pdid) throws WorkflowException
	プロセス定義状態のフェッチ	WMTErrRetType WMFetchProcessDefinitionState (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle, out WMTPProcDefState pproc_def_state)
		不要なので使用しない。
	プロセス定義状態一覧のクローズ	WMTErrRetType WMCloseProcessDefinitionStatesList (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle)
		不要なので使用しない。
5.4.2	プロセス定義状態の変更	WMTErrRetType WMChangeProcessDefinitionState (in WMTPSessionHandle psession_handle, in WMTPProcDefID pproc_def_id, in WMTPProcDefState pproc_def_state)
		クラスにおいて以下のメソッドを提供する。 void changeProcessDefinitionState (Session session, String pdid, ProcessDefinitionState state) throws WorkflowException
7.3.2	プロセスインスタンスの生成	WMTErrRetType WMCreateProcessInstance (in WMTPSessionHandle psession_handle, in WMTPProcDefID pproc_def_id, in WMTPText pproc_inst_name, out WMTPProcInstID pproc_inst_id)
		クラスにおいて以下のメソッドを提供する。 String createProcess (Session session, String pdid, String name) throws WorkflowException
7.3.2	プロセスの開始	WMTErrRetType WMStartProcess (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, out WMTPProcInstID pnew_proc_inst_id)
		クラスにおいて以下のメソッドを提供する。 String startProcess (Session session, String pid) throws WorkflowException

7.3.2	プロセスの停止	WMTErrRetType WMTerminateProcessInstance (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id)
		クラスにおいて以下のメソッドを提供する。 void terminateProcess (Session session, String pid) throws WorkflowException
7.3.2	プロセスインスタンス状態一覧のオープン	WMTErrRetType WMOpenProcessInstanceStatesList (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPFilter pproc_inst_state_filter, in WMTBoolean count_flag, out WMTPQueryHandle pquery_handle, out WMTPInt32 pcount)
		クラスにおいて以下のメソッドを提供する。 ProcessState[] getProcessStates (Session session, String pid, Filter filter) throws WorkflowException
	プロセスインスタンス状態のフェッチ	WMTErrRetType WMFetchProcessInstanceState (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle, out WMTPProcInstState pproc_inst_state)
		不要なので使用しない。
	プロセスインスタンス状態一覧のクローズ	WMTErrRetType WMCloseProcessInstanceStatesList (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle)
		不要なので使用しない。
7.3.2	プロセスインスタンス状態の変更	WMTErrRetType WMChangeProcessInstanceState (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPProcInstState pproc_inst_state)
		クラスにおいて以下のメソッドを提供する。 void changeProcessState(Session session, String pid, ProcessState state) throws WorkflowException
7.3.2	プロセスインスタンス属性一覧のオープン	WMTErrRetType WMOpenProcessInstanceAttributesList (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPFilter pproc_inst_attr_filter, in WMTBoolean count_flag, out WMTPQueryHandle pquery_handle, out WMTPInt32 pcount)
		クラスにおいて以下のメソッドを提供する。 Attribute[] getProcessAttributes(Session session, String pid, Filter filter) throws WorkflowException
	プロセスインスタンス属性のフェッチ	WMTErrRetType WMFetchProcessInstanceAttribute (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle, out WMTPAttrName pattribute_name, out WMTPInt32 pattribute_type, out WMTPInt32 pattribute_length, out WMTPText pattribute_value, in WMTInt32 buffer_size)

		不要なので使用しない。
	プロセスインスタンス属性一覧のクローズ	WMTErrRetType WMCloseProcessInstanceAttributesList (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle)
		不要なので使用しない。
7.3.2	プロセスインスタンス属性の属性値の取得	WMTErrRetType WMGetProcessInstanceAttributeValue (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPAttrName pattribute_name, out WMTPInt32 pattribute_type, out WMTPInt32 pattribute_length, out WMTPText pattribute_value, in WMTInt32 buffer_size)
		クラスにおいて以下のメソッドを提供する。 Attribute getProcessAttributeValue(Session session, String pid, String name) throws WorkflowException
7.3.2	プロセスインスタンス属性の割り当て	WMTErrRetType WMAssignProcessInstanceAttribute (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPAttrName pattribute_name, in WMTInt32 attribute_type, in WMTInt32 attribute_length, in WMTPText pattribute_value)
		クラスにおいて以下のメソッドを提供する。 void assignProcessAttribute(Session session, String pid, Attribute attr) throws WorkflowException

(3) アクティビティ制御機能

仕様書項番	機能名	仕様書記述の API（C 言語） 本ワークフロー運用基盤の API（Java）
7.3.2	アクティビティインスタンス状態一覧のオープン	WMTErrRetType WMOpenActivityInstanceStatesList (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPActivityInstID pactivity_inst_id, in WMTPFilter pact_inst_state_filter, in WMTBoolean count_flag, out WMTPQueryHandle pquery_handle, out WMTPInt32 pcount)
		クラスにおいて以下のメソッドを提供する。 ActivityState[] getActivityStates(Session session, String pid, String aid, Filter filter) throws WorkflowException
	アクティビティインスタンス状態のフェッチ	WMTErrRetType WMFetchActivityInstanceState (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle, out WMTPActivityInstState pactivity_inst_state)
		不要なので使用しない。
	アクティビティインスタンス状態一覧のクローズ	WMTErrRetType WMClos*-eActivityInstanceStatesList (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle)
		不要なので使用しない。

7.3.2	アクティビティインスタンス状態の変更	WMTErrRetType WMChangeActivityInstanceState (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPActivityInstID pactivity_inst_id, in WMTPActivityInstState pactivity_inst_state)
		クラスにおいて以下のメソッドを提供する。 void changeActivityInstanceState(Session session, String pid, String aid, ActivityState state) throws WorkflowException
7.3.2	アクティビティインスタンス属性一覧のオープン	WMTErrRetType WMOpenActivityInstanceAttributesList (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPActivityInstID pactivity_inst_id, in WMTPFilter pact_inst_attr_filter, in WMTBoolean count_flag, out WMTPQueryHandle pquery_handle, out WMTPInt32 pcount)
		クラスにおいて以下のメソッドを提供する。 Attribute[] getActivityAttributes(Session session, String pid, String aid, Filter filter) throws WorkflowException
	アクティビティインスタンス属性のフェッチ	WMTErrRetType WMFetchActivityInstanceAttribute (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle, out WMTPAttrName pattribute_name, out WMTPInt32 pattribute_type, out WMTPInt32 pattribute_length, out WMTPText pattribute_value, in WMTInt32 buffer_size)
		不要なので使用しない。
	アクティビティインスタンス属性一覧のクローズ	WMTErrRetType WMCloseActivityInstanceAttributesList (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle)
		不要なので使用しない。
7.3.2	アクティビティインスタンス属性の属性値の取得	WMTErrRetType WMGetActivityInstanceAttributeValue (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPActivityInstID pactivity_inst_id, in WMTPAttrName pattribute_name, out WMTPInt32 pattribute_type, out WMTPInt32 pattribute_length, out WMTPText pattribute_value, in WMTInt32 buffer_size)
		クラスにおいて以下のメソッドを提供する。 Attribute getActivityAttributeValue(Session session, String pid, String aid, String name) throws WorkflowException

7.3.2	アクティビティインスタンス属性の割り当て	WMTErrRetType WMAssignActivityInstanceAttribute (in WMTPSessionHandle psession_handle, in WMTPProcDefID pproc_def_id, in WMTPActivityInstID pactivity_inst_id, in WMTPAttrName pattribute_name, in WMTInt32 attribute_type, in WMTInt32 attribute_length, in WMTPText pattribute_value)
		クラスにおいて以下のメソッドを提供する。 void assignActivityAttribute(Session session, String pid, String aid, Attribute attr) throws WorkflowException

(4) プロセス状態関連機能

仕様書項番	機能名	仕様書記述の API（C 言語） 本ワークフロー運用基盤の API（Java）
7.3.2	プロセスインスタンス一覧のオープン	WMTErrRetType WMOpenProcessInstancesList (in WMTPSessionHandle psession_handle, in WMTPFilter pproc_inst_filter, in WMTBoolean count_flag, out WMTPQueryHandle pquery_handle, out WMTPInt32 pcount)
		クラスにおいて以下のメソッドを提供する。 Process[] getProcesses(Session session, Filter filter) throws WorkflowException
	プロセスインスタンスのフェッチ	WMTErrRetType WMFetchProcessInstance (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle, out WMTPProcInst pproc_inst_buf_ptr)
		不要なので使用しない。
	プロセスインスタンス一覧のクローズ	WMTErrRetType WMCloseProcessInstancesList (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle)
		不要なので使用しない。
7.3.2	プロセスインスタンスの取得	WMTErrRetType WMGetProcessInstance (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, out WMTPProcInst pproc_inst)
		クラスにおいて以下のメソッドを提供する。 Process getProcess(Session session, String pid) throws WorkflowException

(5) アクティビティ状態関連機能

仕様書項番	機能名	仕様書記述の API（C 言語） 本ワークフロー運用基盤の API（Java）
7.3.2	アクティビティインスタンス一覧のオープン	WMTErrRetType WMOpenActivityInstancesList (in WMTPSessionHandle psession_handle, in WMTPFilter pactivity_inst_filter, in WMTBoolean count_flag, out WMTPQueryHandle pquery_handle, out WMTPInt32 pcount)

		クラスにおいて以下のメソッドを提供する。 Activity[] getActivities(Session session, Filter filter) 　throws WorkflowException
	アクティビティインスタンスのフェッチ	WMTErrRetType WMFetchActivityInstance (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle, out WMTPActivityInst pactivity_inst)
		不要なので使用しない。
	アクティビティインスタンス一覧のクローズ	WMTErrRetType WMCloseActivityInstancesList (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle)
		不要なので使用しない。
7.3.2	アクティビティインスタンスの取得	WMTErrRetType WMGetActivityInstance (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPActivityInstID pactivity_inst_id, out WMTPActivityInst pactivity_inst)
		クラスにおいて以下のメソッドを提供する。 Activity getActivity(Session session, String pid, String aid) 　throws WorkflowException

(6) 作業項目関連機能

仕様書 項番	機能名	仕様書記述の API（C 言語）
		本ワークフロー運用基盤の API（Java）
7.4.2	作業項目一覧のオープン	WMTErrRetType WMOpenWorkList (in WMTPSessionHandle psession_handle, in WMTPFilter pworklist_filter, in WMTBoolean count_flag, out WMTPQueryHandle pquery_handle, out WMTPInt32 pcount)
		クラスにおいて以下のメソッドを提供する。 WorkItem[] getWorkItems(Session session, Filter filter) 　throws WorkflowException
	作業項目のフェッチ	WMTErrRetType WMFetchWorkItem (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle, out WMTPWorkItem pwork_item)
		不要なので使用しない。
	作業項目一覧のクローズ	WMTErrRetType WMCloseWorkList (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle)
		不要なので使用しない。
7.4.2	作業項目の取得	WMTErrRetType WMGetWorkItem (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPWorkItemID pwork_item_id, out WMTPWorkItem pwork_item)
		クラスにおいて以下のメソッドを提供する。 WorkItem getWorkItem(Session session, String wid) 　throws WorkflowException

7.4.2	作業項目の完了	WMTErrRetType WMCompleteWorkItem (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPWorkItemID pwork_item_id)
		クラスにおいて以下のメソッドを提供する。 void completeWorkItem(Session session, String wid) throws WorkflowException
7.4.2	作業項目状態一覧のオープン	WMTErrRetType WMOpenWorkitemStatesList (in WMTPSessionHandle psession_handle, in WMTPProcDefID pproc_def_id, in WMTPFilter pworkitem_state_filter, in WMTBoolean count_flag, out WMTPQueryHandle pquery_handle, out WMTUInt32 pcount)
		クラスにおいて以下のメソッドを提供する。 WorkItemState[] getWorkItemStates(Session session, String pdid, Filter filter) throws WorkflowException
	作業項目状態のフェッチ	WMTErrRetType WMFetchWorkitemState (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle, out WMTPProcDefState pworkitem_state)
		不要なので使用しない。
	作業項目状態一覧のクローズ	WMTErrRetType WMCloseWorkitemStatesList (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle)
		不要なので使用しない。
7.4.2	作業項目状態の変更	WMTErrRetType WMChangeWorkitemState (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPWorkItemID pwork_item_id, in WMTPProcDefState pworkitem_state)
		クラスにおいて以下のメソッドを提供する。 WorkItemState[] changeWorkItemState (Session session, String wid, ProcessDefinitionState state) throws WorkflowException
7.4.2	作業項目の再割り当て	WMTErrRetType WMReassignWorkItem (in WMTPSessionHandle psession_handle, in WMTPWflParticipant psource_user, in WMTPWflParticipant ptarget_user, in WMTPProcInstID pproc_inst_id, in WMTPWorkItemID pwork_item_id)
		クラスにおいて以下のメソッドを提供する。 void reassignWorkItem (Session session, WfResource sourceUser, WfResource targetUser, String wid) throws WorkflowException

7.4.2	作業項目属性一覧のオープン	WMTErrRetType WMOpenWorkItemAttributesList (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPWorkItemID pwork_item_id, in WMTPFilter pwork_item_attr_filter, in WMTBoolean count_flag, out WMTPQueryHandle pquery_handle, out WMTPInt32 pcount)
		クラスにおいて以下のメソッドを提供する。 Attribute[] getWorkItemAttributes(Session session, String wid, Filter filter) throws WorkflowException
	作業項目属性のフェッチ	WMTErrRetType WMFetchWorkItemAttribute (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle, out WMTPAttrName pattribute_name, out WMTPInt32 pattribute_type, out WMTPInt32 pattribute_length, out WMTPText pattribute_value, in WMTInt32 buffer_size)
		不要なので使用しない。
	作業項目属性一覧のクローズ	WMTErrRetType WMCloseWorkItemAttributesList (in WMTPSessionHandle psession_handle, in WMTPQueryHandle pquery_handle)
		不要なので使用しない。
7.4.2	作業項目属性の取得	WMTErrRetType WMGetWorkItemAttributeValue (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPWorkItemID pwork_item_id, in WMTPAttrName pattribute_name, out WMTPInt32 pattribute_type, out WMTPInt32 pattribute_length, out WMTPText pattribute_value, in WMTInt32 buffer_size)
		クラスにおいて以下のメソッドを提供する。 Attribute getWorkItemAttribute(Session session, String wid, String name) throws WorkflowException
7.4.2	作業項目属性の割り当て	WMTErrRetType WMAssignWorkItemAttribute (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id, in WMTPWorkItemID pwork_item_id, in WMTPAttrName pattribute_name, in WMTInt32 attribute_type, in WMTInt32 attribute_length, in WMTPText pattribute_value)
		クラスにおいて以下のメソッドを提供する。 void assignWorkItemAttribute(Session session, String wid, Attribute attr) throws WorkflowException

(7) 管理関連機能

仕様書 項番	機能名	仕様書記述の API（C 言語）
		本ワークフロー運用基盤の API（Java）

7.5.2	プロセスインスタンス状態の変更	WMTErrRetType WMChangeProcessInstancesState (in WMTPSessionHandle psession_handle, in WMTPProcDefID pproc_def_id, in WMTPFilter pproc_inst_filter, in WMTPProcInstState pproc_inst_state)
		クラスにおいて以下のメソッドを提供する。 void changeProcessesState(Session session, String pdid, Filter filter, ProcessState state) throws WorkflowException
7.5.2	アクティビティインスタンス状態の変更	WMTErrRetType WMChangeActivityInstancesState (in WMTPSessionHandle psession_handle, in WMTPProcDefID pproc_def_id, in WMTPActivityID pactivity_def_id, in WMTPFilter pact_inst_filter, in WMTPActivityInstState pactivity_inst_state)
		クラスにおいて以下のメソッドを提供する。 void changeActivitiesState(Session session, String pdid, String adid, Filter filter, ActivityState state) throws WorkflowException
7.5.2	プロセスインスタンスの停止	WMTErrRetType WMTerminateProcessInstances (in WMTPSessionHandle psession_handle, in WMTPProcDefID pproc_def_id, in WMTPFilter pproc_inst_filter)
		クラスにおいて以下のメソッドを提供する。 void terminateProcesses(Session session, String pdid, Filter filter) throws WorkflowException
7.5.2	プロセスインスタンス属性の割り当て	WMTErrRetType WMAssignProcessInstancesAttribute (in WMTPSessionHandle psession_handle, in WMTPProcDefID pproc_def_id, in WMTPFilter pproc_inst_filter, in WMTPAttrName pattribute_name, in WMTInt32 attribute_type, in WMTInt32 attribute_length, in WMTPText pattribute_value)
		クラスにおいて以下のメソッドを提供する。 void assignProcessesAttribute(Session session, String pdid, Filter filter, Attribute attr) throws WorkflowException
7.5.2	アクティビティインスタンス属性の割り当て	WMTErrRetType WMAssignActivityInstancesAttribute (in WMTPSessionHandle psession_handle, in WMTPProcDefID pproc_def_id, in WMTPActivityID pactivity_def_id, in WMTPFilter pact_inst_filter, in WMTPAttrName pattribute_name, in WMTInt32 attribute_type, in WMTInt32 attribute_length, in WMTPText pattribute_value)

		クラスにおいて以下のメソッドを提供する。 void assignActivitiesAttribute(Session session, String pdid, String adid, Filter filter, Attribute attr) throws WorkflowException
7.5.2	複数のプロセスインスタンスの中断	WMTErrRetType WMAbortProcessInstances (in WMTPSessionHandle psession_handle, in WMTPProcDefID pproc_def_id, in WMTPFilter pproc_inst_filter)
		クラスにおいて以下のメソッドを提供する。 void abortProcesses(Session session, String pdid, Filter filter) throws WorkflowException
7.5.2	プロセスインスタンスの中断	WMTErrRetType WMAbortProcessInstance (in WMTPSessionHandle psession_handle, in WMTPProcInstID pproc_inst_id)
		クラスにおいて以下のメソッドを提供する。 void abortProcess(Session session, String pid) throws WorkflowException

(8) アプリケーション起動関連機能（ツールエージェント）

仕様書 項番	機能名	仕様書記述の API（C 言語） 本ワークフロー運用基盤の API（Java）
	ツールエージェントへの接続	WMTAConnect()
		不要なので使用しない。 かわりに、invokeApplication の戻り値で、ツールエージェントハンドル ID が返るものとする。
	ツールエージェントとの切断	WMTADisconnect()
		不要なので使用しない。
8.3.2	アプリケーションの起動	int WMTAInvokeApplication (in int tool_agent_handle, in string application_name, in WMTPProcInstID pproc_inst_id, in WMTPWorkItemID pwork_item_id, in WMTPAttributeList pattribute_list, in int app_mode)
		ToolAgent インタフェースにおいて以下のメソッドを提供する。 int invokeApplication(String appName, String wid, Attribute[] attrList, int mode) throws WorkflowException
8.3.2	アプリケーション状態の要求	int WMTARequestAppStatus (in WMTInt32 tool_agent_handle, in WMTPProcInstID proc_inst_id, in WMTPWorkItemID pwork_item_id, out WMTInt32 app_status, out WMTPAttributeList WFRelevantData)

		ToolAgent インタフェースにおいて以下のメソッドを提供する。 int requestAppStatus(int toolAgentHandle, String appName, String wid, Attribute[] attrList) throws WorkflowException
8.3.2	アプリケーションの停止	int WMTATerminateApp （ in WMTInt32 tool_agent_handle, in WMTPProcInstID pproc_inst_id, in WMTPWorkItemID pwork_item_id)
		ToolAgent インタフェースにおいて以下のメソッドを提供する。 void terminateApplication(int toolAgentHandle, String wid) throws WorkflowException

4.3 インタフェース4

仕様書 項番	機能名	仕様書記述の API（引数情報） 本ワークフロー運用基盤の API（Java）
9.3.2	プロセスインスタンス状態の変更	Engine identifier Process id State
		クラスにおいて以下のメソッドを提供する。 void changeProcessState(String eid, String pid, ProcessState state) throws WorkflowException
9.3.2	プロセスインスタンスの生成	Engine identifier Process definition id Return flag Parent pid Activity id Sub-process id User id Role id
		クラスにおいて以下のメソッドを提供する。 String createProcess(String eid, String pdid, int flag, String parentid, String aid, String userid, String roleid) throws WorkflowException
9.3.2	プロセスインスタンス属性の取得	Engine identifier Process id Root pid Activity id Attributes
		クラスにおいて以下のメソッドを提供する。 Attribute[] getProcessAttributes(String eid, String pid, String rootid, String aid) throws WorkflowException

9.3.2	プロセスインスタンス状態の取得	Engine identifier Process id State
		クラスにおいて以下のメソッドを提供する。 ProcessState getProcessState(String eid, String pid) throws WorkflowException
9.3.2	プロセスインスタンス属性の変更通知	Engine identifier Process id Attributes
		クラスにおいて以下のメソッドを提供する。 void processAttributesChanged(String eid, String pid, Attribute[] attr) throws WorkflowException
9.3.2	プロセスインスタンス状態の変更通知	Engine identifier Process id New State
		クラスにおいて以下のメソッドを提供する。 void processStateChanged(String eid, String pid, ProcessState state) throws WorkflowException
9.3.2	プロセスインスタンス属性の設定	Engine identifier Root pid Activity id Process id Attributes
		クラスにおいて以下のメソッドを提供する。 void setProcessAttributes(String eid, String rootid, String aid, String pid, Attrbute[] attr) throws WorkflowException
9.3.2	アクティビティに対するトリガー	Engine identifier Root pid Activity id Process id
		クラスにおいて以下のメソッドを提供する。 void triggerActivity(String eid, String rootid, String aid, String pid) throws WorkflowException
9.3.2	プロセスインスタンス一覧の取得	Engine identifier Filter
		クラスにおいて以下のメソッドを提供する。 ProcessID[] listProcesses(String eid, Filter filter) throws WorkflowException
9.3.2	プロセスインスタンスの放棄	Engine identifier Process id

		クラスにおいて以下のメソッドを提供する。 void relinquishProcess(String eid, String pid) throws WorkflowException

4.4 インタフェース5

4.4.1 監査データ型定義

仕様書 項番	機能名	仕様書記述の API（C 言語） 本ワークフロー運用基盤の API（Java）
	監査情報の長さ	typedef { WMTInt16 information_length; } WMAInformationLength; 不要なので使用しない。
	監査情報型	typedef { WMTText information_type; } WMAInformationType;
	監査対象のオブジェクト ID	typedef { WMTText object_id[UNIQUE_ID_SIZE]; } WMAObjectID; String 型とする。

	監査対象のオブジェクト状態	typedef {
		WMTText object_state;
		} WMAObjectState;
		状態の基底クラスとして、以下のインタフェースを提供する。
		public interface WMObjectState {
		}

	監査対象のオブジェクト名	typedef { WMTText object_name[NAME_STRING_SIZE]; } WMAObjectName; String 型とする。
	監査対象のリソース ID	typedef { WMTText resource_id[UNIQUE_ID_SIZE]; } WMAResourceID; String 型とする。
	監査対象のリソース名	typedef { WMTText resource_name[NAME_STRING_SIZE]; } WMAResourceName; String 型とする。
	監査対象の属性名	typedef { WMTText attribute_name[NAME_STRING_SIZE]; } WMAAttributeName; String 型とする。

	監査対象の属性型	typedef {
		WMTInt32 attribute_type;
		} WMAAttributeType;
		int 型とし、以下の値を WMWorkflowConstants インタフェースにて提供する。
		static final int ATTR_STRING_TYPE = 1;
		static final int ATTR_INTEGER_TYPE = 2;
		static final int ATTR_BOOLEAN_TYPE = 3;

	監査対象の属性値の長さ	typedef { WMTInt32 attribute_length; } WMAAttributeLength; 不要なので使用しない。
	監査対象の属性値	typedef { WMTText attribute_value[NAME_STRING_SIZE]; } WMAAttributeValue; String 型とする。

	監査対象のセッション ID	typedef { WMTText session_id[UNIQUE_ID_SIZE]; } WMASessionID;
		String 型とする。
	コードページ	typedef { WMTInt16 code_page; } WMACodePage;
		不要なので使用しない。
	タイムスタンプ	typedef { WMTText timestamp[TIME_STAMP_SIZE]; } WMATimeStamp;
		String 型とし、”YYYY/MM/DD hh:mm:ss”形式を採用する。
	監査イベントコード	typedef { WMTText event_code; } WMAEventCode;
		String 型とする。

4.4.2 CWAD 接頭情報

仕様書 項番	機能名	仕様書記述の API（監査情報）
		本ワークフロー運用基盤の API（Java）
	CWAD 接頭情報	InitialProcessInstanceID M WMAObjectID CurrentProcessInstanceID M WMAObjectID ActivityInstanceID O WMAObjectID ProcessState M WMAObjectState EventCode M WMAEventCode DomainID M WMAResourceID NodeID M WMAResourceID UserID O* WMAResourceID RoleID O* WMAResourceID Timestamp M WMATimestamp InformationID M WMAObjectID
		型マッピングは、データ型定義に合わせる。

4.4.3 CWAD 接尾情報

仕様書 項番	機能名	仕様書記述の API（監査情報）
		本ワークフロー運用基盤の API（Java）
	CWAD 接尾情報	AccountCode O WMAInformationType ExtensionNumber M WMTInt16 ExtensionType M WMAInformationType ExtensionLength O WMAInformationLength ExtensionCodePage O WMACodePage ExtensionContent O *
		型マッピングは、データ型定義に合わせる。

4.4.4 プロセスインスタンス監査情報

仕様書 項番	機能名	仕様書記述の API（監査情報）
		本ワークフロー運用基盤の API（Java）
	プロセス/サブプロセスインスタンスの生成/開始監査	Prefix M ---- ProcessDefinitionID M WMAObjectID ProcessDefinitionBusinessName O WMAObjectName Suffix M ----
		型マッピングは、データ型定義に合わせる。
	プロセスインスタンス状態の変更監査	Prefix M ---- PreviousProcessState M WMAObjectState NewProcessState M WMAObjectState Suffix M ----
		型マッピングは、データ型定義に合わせる。

	プロセスインスタンス属性の割り当て監査	Prefix M ---- AttributeName M WMAAttributeName AttributeType M WMAAttributeType NewAttributeLength M WMAAttributeLength NewAttributeValue M WMAAttributeValue PreviousAttributeLength M WMAAttributeLength PreviousAttributeValue M WMAAttributeValue Suffix M ---- 型マッピングは、データ型定義に合わせる。

4.4.5 アクティビティインスタンス監査情報

仕様書項番	機能名	仕様書記述の API（監査情報） 本ワークフロー運用基盤の API（Java）
	アクティビティインスタンス状態の変更監査	Prefix M ---- ActivityInstanceID M WMAObjectID ActivityDefinitionBusinessName O WMAObjectName ApplicationID O WMAObjectID NewActivityState M WMAObjectState PreviousActivityState M WMAObjectState Suffix M ---- 型マッピングは、データ型定義に合わせる。
	アクティビティインスタンス属性の割り当て監査	Prefix M ---- ActivityInstanceID M WMAObjectID ActivityState O WMAObjectState AttributeName M WMAAttributeName AttributeType M WMAAttributeType NewAttributeLength M WMAAttributeLength NewAttributeValue M WMAAttributeValue PreviousAttributeLength M WMAAttributeLength PreviousAttributeValue M WMAAttributeValue Suffix M ---- 型マッピングは、データ型定義に合わせる。

4.4.6 作業項目監査情報

仕様書項番	機能名	仕様書記述の API（監査情報） 本ワークフロー運用基盤の API（Java）
	作業項目状態の変更監査	Prefix M ---- ActivityInstanceID M WMAObjectID WorkItemID M WMAObjectID WorkItemState M WMAObjectState Suffix M ---- 型マッピングは、データ型定義に合わせる。
	作業項目の割り当て/再割り当て監査	Prefix M ---- ActivityInstanceID M WMAObjectID WorkItemID M WMAObjectID WorkItemState M WMAObjectState TargetDomainID M WMAResourceID TargetNodeID M WMAResourceID TargetUserID O WMAResourceID TargetRoleID O WMAResourceID Suffix M ---- 型マッピングは、データ型定義に合わせる。
	作業項目属性の割り当て監査	Prefix M ---- ActivityInstanceID M WMAObjectID ActivityState O WMAObjectState AttributeName M WMAAttributeName AttributeType M WMAAttributeType NewAttributeLength M WMAAttributeLength NewAttributeValue M WMAAttributeValue PreviousAttributeLength M WMAAttributeLength PreviousAttributeValue M WMAAttributeValue Suffix M ----

		型マッピングは、データ型定義に合わせる。

4.4.7 リモート操作情報

(1) 会話の開始

仕様書 項番	機能名	仕様書記述の API（監査情報） 本ワークフロー運用基盤の API（Java）
	要求元エンジンの監査	Prefix M ---- MessageID O WMAObjectID CorrespondentContextID M WMAResourceID CorrespondentNodeID M WMAResourceID Extension Number M WMTInt16 Extension Type M WMTText SourceConverstaion ID M WMTText TargetConversation ID M WMTText 型マッピングは、データ型定義に合わせる。
	要求先エンジンの監査	Prefix M ---- MessageID O WMAObjectID CorrespondentContextID M WMAResourceID CorrespondentNodeID M WMAResourceID Extension Number M WMTInt16 Extension Type M WMTText SourceConverstaion ID M WMTText TargetConversation ID M WMTText 型マッピングは、データ型定義に合わせる。

(2) 会話の停止

仕様書 項番	機能名	仕様書記述の API（監査情報） 本ワークフロー運用基盤の API（Java）
	要求元エンジンの監査	Prefix M ---- MessageID O WMAObjectID CorrespondentContextID M WMAResourceID CorrespondentNodeID M WMAResourceID Extension Number M WMTInt16 Extension Type M WMTText SourceConverstaion ID M WMTText TargetConversation ID M WMTText 型マッピングは、データ型定義に合わせる。
	要求先エンジンの監査	Prefix M ---- MessageID O WMAObjectID CorrespondentContextID M WMAResourceID CorrespondentNodeID M WMAResourceID Extension Number M WMTInt16 Extension Type M WMTText SourceConverstaion ID M WMTText TargetConversation ID M WMTText 型マッピングは、データ型定義に合わせる。

(3) プロセスインスタンスの生成

仕様書 項番	機能名	仕様書記述の API（監査情報） 本ワークフロー運用基盤の API（Java）

	要求元エンジンの要求監査	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceActivityDefinition-BusinessName	O	WMAObjectName
		TargetProcessDefinition ID	M	WMAObjectID
		TargetProcessInstance ID	M	WMAObjectID
		TargetProcessDefinition-BusinessName	O	WMAObjectName
		TargetNodeID	M	WMTResourceID
		TargetUserID	O	WMTResourceID
		TargetRoleID	O	WMTResourceID
		TargetState	O	WMTObjectState
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		
	要求先エンジンの要求監査	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceInitialProcessInstanceID	M	WMAObjectID
		SourceCurrentProcessInstanceID	M	WMAObjectID
		SourceActivityInstanceID	M	WMAObjectID
		SourceTimestamp	M	WMTDateTime
		SourceNodeID	M	WMTResourceID
		SourceUserID	O	WMTResourceID
		SourceRoleID	O	WMTResourceID
		SourceProcessDefinitionID	M	WMTObjectID
		SourceProcessDefinition-BusinessName	O	WMTObjectName
		SourceActivityDefinition-BusinessName	O	WMTObjectName
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		
	要求元エンジンの操作監査	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceActivityInstanceID	M	WMAObjectID
		RemoteNodeID	M	WMAResourceID
		RemoteProcessInstanceID	M	WMAObjectID
		RemoteTimestamp	M	WMTDateTime
		RemoteProcessDefinition-BusinessName	O	WMTObjectName
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		
	要求先エンジンの操作監査	Prefix	M	----
		MessageID	O	WMAObjectID
		ProcessDefinitionID	M	WMAObjectID
		ProcessDefinitionBusinessName	O	WMTObjectName
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		

	要求元エンジンの応答監査	Prefix M ---- MessageID O WMAObjectID SourceActivityDefinition-BusinessName O WMAObjectName TargetProcessDefinition ID M WMAObjectID TargetProcessInstance ID M WMAObjectID TargetProcessDefinition-BusinessName O WMAObjectName TargetNodeID M WMTResourceID TargetUserID O WMTResourceID TargetRoleID O WMTResourceID TargetState O WMTObjectState Extension Number M WMTInt16 Extension Type M WMTText SourceConverstaion ID M WMTText TargetConversation ID M WMTText
		型マッピングは、データ型定義に合わせる。
	要求先エンジンの応答監査	Prefix M ---- MessageID O WMAObjectID SourceInitialProcessInstanceID M WMAObjectID SourceCurrentProcessInstanceID M WMAObjectID SourceActivityInstanceID M WMAObjectID SourceTimestamp M WMTDateTime SourceNodeID M WMTResourceID SourceUserID O WMTResourceID SourceRoleID O WMTResourceID SourceProcessDefinitionID M WMTObjectID SourceProcessDefinition-BusinessName O WMTObjectName SourceActivityDefinition-BusinessName O WMTObjectName Extension Number M WMTInt16 Extension Type M WMTText SourceConverstaion ID M WMTText TargetConversation ID M WMTText
		型マッピングは、データ型定義に合わせる。

(4) プロセスインスタンス状態の変更

仕様書項番	機能名	仕様書記述の API（監査情報）
		本ワークフロー運用基盤の API（Java）
	要求元エンジンの要求監査	Prefix M ---- MessageID O WMAObjectID SourceActivityDefinition-BusinessName O WMAObjectName TargetProcessDefinition ID O WMAObjectID TargetProcessInstance ID M WMAObjectID TargetProcessDefinition-BusinessName O WMAObjectName TargetNodeID M WMTResourceID TargetUserID O WMTResourceID TargetRoleID O WMTResourceID TargetState O WMTObjectState Extension Number M WMTInt16 Extension Type M WMTText SourceConverstaion ID M WMTText TargetConversation ID M WMTText
		型マッピングは、データ型定義に合わせる。

	要求先エンジンの要求監査	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceInitialProcessInstanceID	M	WMAObjectID
		SourceCurrentProcessInstanceID	M	WMAObjectID
		SourceActivityInstanceID	M	WMAObjectID
		SourceTimestamp	M	WMTDateTime
		SourceNodeID	M	WMTResourceID
		SourceUserID	O	WMTResourceID
		SourceRoleID	O	WMTResourceID
		SourceProcessDefinitionID	M	WMTObjectID
		SourceProcessDefinition-BusinessName	O	WMTObjectName
		SourceActivityDefinition-BusinessName	O	WMTObjectName
		SourceRequestedState	M	WMTObjectState
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		
	要求元エンジンの操作監査	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceActivityInstanceID	M	WMAObjectID
		RemoteNodeID	M	WMAResourceID
		RemoteProcessInstanceID	M	WMAObjectID
		RemoteTimestamp	M	WMTDateTime
		RemoteProcessDefinition-BusinessName	O	WMTObjectName
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		
	要求先エンジンの操作監査	Prefix	M	----
		MessageID	O	WMAObjectID
		PreviousProcessState	M	WMAObjectState
		NewProcessState	M	WMTObjectState
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		
	要求元エンジンの応答監査	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceInitialProcessInstanceID	M	WMAObjectID
		SourceCurrentProcessInstanceID	M	WMAObjectID
		SourceActivityInstanceID	M	WMAObjectID
		SourceTimestamp	M	WMTDateTime
		SourceNodeID	M	WMTResourceID
		SourceUserID	O	WMTResourceID
		SourceRoleID	O	WMTResourceID
		SourceProcessDefinitionID	O	WMTObjectID
		SourceProcessDefinition-BusinessName	O	WMTObjectName
		SourceActivityDefinition-BusinessName	O	WMAObjectName
		SourceRequestedState	M	WMTObjectState
		NewState	M	WMTObjectState
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText

		型マッピングは、データ型定義に合わせる。
	要求先エンジンの応答監査	Prefix M ---- MessageID O WMAObjectID SourceActivityDefinition-BusinessName O WMAObjectName TargetProcessDefinition ID O WMAObjectID TargetProcessInstance ID O WMAObjectID TargetProcessDefinition-BusinessName O WMAObjectName TargetNodeID M WMTResourceID TargetUserID O WMTResourceID TargetRoleID O WMTResourceID TargetState M WMTObjectState PreviousState M WMTObjectState Extension Number M WMTInt16 Extension Type M WMTText SourceConverstaion ID M WMTText TargetConversation ID M WMTText
		型マッピングは、データ型定義に合わせる。

(5) プロセスインスタンス属性の取得

仕様書項番	機能名	仕様書記述の API（監査情報）
		本ワークフロー運用基盤の API（Java）
	要求元エンジンの要求監査	Prefix M ---- MessageID O WMAObjectID SourceActivityDefinition-BusinessName O WMAObjectName TargetProcessInstance ID M WMAObjectID TargetProcessDefinition-BusinessName O WMAObjectName TargetNodeID M WMTResourceID TargetUserID O WMTResourceID TargetRoleID O WMTResourceID AttributeName M WMTAttributeName Extension Number M WMTInt16 Extension Type M WMTText SourceConverstaion ID M WMTText TargetConversation ID M WMTText
		型マッピングは、データ型定義に合わせる。
	要求先エンジンの要求監査	Prefix M ---- MessageID O WMAObjectID SourceInitialProcessInstanceID M WMAObjectID SourceCurrentProcessInstanceID M WMAObjectID SourceActivityInstanceID M WMAObjectID SourceTimestamp M WMTDateTime SourceNodeID M WMTResourceID SourceUserID O WMTResourceID SourceRoleID O WMTResourceID SourceProcessDefinitionID M WMTObjectID SourceProcessDefinition-BusinessName O WMTObjectName SourceActivityDefinition-BusinessName O WMTObjectName AttributeName M WMTAttributeName Extension Number M WMTInt16 Extension Type M WMTText SourceConverstaion ID M WMTText TargetConversation ID M WMTText
		型マッピングは、データ型定義に合わせる。

	要求元エンジンの応答監査	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceInitialProcessInstanceID	M	WMAObjectID
		SourceCurrentProcessInstanceID	M	WMAObjectID
		SourceActivityInstanceID	M	WMAObjectID
		SourceTimestamp	M	WMTDateTime
		SourceNodeID	M	WMTResourceID
		SourceUserID	O	WMTResourceID
		SourceRoleID	O	WMTResourceID
		SourceProcessDefinitionID	M	WMTObjectID
		SourceProcessDefinition-BusinessName	O	WMTObjectName
		SourceActivityDefinition-BusinessName	O	WMTObjectName
		AttributeName	O	WMTAttributeName
		AttributeType	O	WMTAttributeType
		Attributelength	O	WMTAttributeLength
		Attributevalue	O	WMTAttributeValue
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		
	要求先エンジンの応答監査	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceActivityDefinition-BusinessName	O	WMAObjectName
		TargetProcessDefinition ID	O	WMAObjectID
		TargetProcessInstance ID	O	WMAObjectID
		TargetProcessDefinition-BusinessName	O	WMAObjectName
		TargetNodeID	M	WMTResourceID
		TargetUserID	O	WMTResourceID
		TargetRoleID	O	WMTResourceID
		AttributeName	O	WMTAttributeName
		AttributeType	O	WMTAttributeType
		Attributelength	O	WMTAttributeLength
		Attributevalue	O	WMTAttributeValue
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		

(6) プロセスインスタンス属性の設定

仕様書 項番	機能名	仕様書記述の API（監査情報）
		本ワークフロー運用基盤の API（Java）

	要求元エンジンの要求監査	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceActivityDefinition-BusinessName	O	WMAObjectName
		TargetProcessDefinition ID	O	WMAObjectID
		TargetProcessInstance ID	M	WMAObjectID
		TargetProcessDefinition-BusinessName	O	WMAObjectName
		TargetNodeID	M	WMTResourceID
		TargetUserID	O	WMTResourceID
		TargetRoleID	O	WMTResourceID
		AttributeName	O	WMTAttributeName
		AttributeType	O	WMTAttributeType
		Attributelength	O	WMTAttributeLength
		Attributevalue	O	WMTAttributeValue
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		
	要求先エンジンの要求監査	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceInitialProcessInstanceID	M	WMAObjectID
		SourceCurrentProcessInstanceID	M	WMAObjectID
		SourceActivityInstanceID	M	WMAObjectID
		SourceTimestamp	M	WMTDateTime
		SourceNodeID	M	WMTResourceID
		SourceUserID	O	WMTResourceID
		SourceRoleID	O	WMTResourceID
		SourceProcessDefinitionID	M	WMTObjectID
		SourceProcessDefinition-BusinessName	O	WMTObjectName
		SourceActivityDefinition-BusinessName	O	WMTObjectName
		AttributeName	O	WMTAttributeName
		AttributeType	O	WMTAttributeType
		Attributelength	O	WMTAttributeLength
		Attributevalue	O	WMTAttributeValue
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		
	要求元エンジンの操作監査	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceActivityInstanceID	M	WMAObjectID
		RemoteNodeID	M	WMAResourceID
		RemoteProcessInstanceID	M	WMAObjectID
		RemoteTimestamp	M	WMTDateTime
		RemoteProcessDefinition-BusinessName	O	WMTObjectName
		AttributeName	O	WMTAttributeName
		AttributeType	O	WMTAttributeType
		Attributelength	O	WMTAttributeLength
		Attributevalue	O	WMTAttributeValue
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		

	要求先エンジンの操作監査	Prefix	M	----
		MessageID	O	WMAObjectID
		AttributeName	O	WMTAttributeName
		AttributeType	O	WMTAttributeType
		Attributelength	O	WMTAttributeLength
		Attributevalue	O	WMTAttributeValue
		PreviousAttributeLength	O	WMTAttributeLength
		PreviousAttributeValue	O	WMTAttributeValue
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		
	要求元エンジンの応答監査	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceInitialProcessInstanceID	M	WMAObjectID
		SourceCurrentProcessInstanceID	M	WMAObjectID
		SourceActivityInstanceID	M	WMAObjectID
		SourceTimestamp	M	WMTDateTime
		SourceNodeID	M	WMTResourceID
		SourceUserID	O	WMTResourceID
		SourceRoleID	O	WMTResourceID
		SourceProcessDefinitionID	O	WMTObjectID
		SourceProcessDefinition-BusinessName	O	WMTObjectName
		SourceActivityDefinition-BusinessName	O	WMTObjectName
		AttributeName	O	WMTAttributeName
		AttributeType	O	WMTAttributeType
		Attributelength	O	WMTAttributeLength
		Attributevalue	O	WMTAttributeValue
		PreviousAttributeLength	O	WMTAttributeLength
		PreviousAttributeValue	O	WMTAttributeValue
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		
	要求先エンジンの応答監査	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceActivityDefinition-BusinessName	O	WMAObjectName
		TargetProcessDefinition ID	O	WMAObjectID
		TargetProcessInstance ID	O	WMAObjectID
		TargetProcessDefinition-BusinessName	O	WMAObjectName
		TargetNodeID	M	WMTResourceID
		TargetUserID	O	WMTResourceID
		TargetRoleID	O	WMTResourceID
		AttributeName	O	WMTAttributeName
		AttributeType	O	WMTAttributeType
		Attributelength	O	WMTAttributeLength
		Attributevalue	O	WMTAttributeValue
		PreviousAttributeLength	O	WMTAttributeLength
		PreviousAttributeValue	O	WMTAttributeValue
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		

(7) プロセスインスタンス状態の変更通知

仕様書	機能名	仕様書記述の API（監査情報）

項番		本ワークフロー運用基盤の API（Java）		
	エンジンへの実イベント監査の通知	Prefix	M	----
		MessageID	O	WMAObjectID
		PreviousProcessState	M	WMTProcInstState
		New ProcessState	M	WMTProcInstState
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		
	エンジンへの通知監査の通知	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceActivityDefinition-BusinessName	O	WMAObjectName
		TargetProcessDefinition ID	O	WMAObjectID
		TargetProcessInstance ID	O	WMAObjectID
		TargetNodeID	M	WMTResourceID
		New ProcessState	M	WMTProcInstState
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		
	要求先エンジンの監査	Prefix	M	----
		MessageID	O	WMAObjectID
		SourceActivityDefinition-BusinessName	O	WMAObjectName
		SourceProcessDefinition ID	M	WMAObjectID
		SourceProcessInstance ID	M	WMAObjectID
		SourceNodeID	M	WMTResourceID
		New ProcessState	M	WMTProcInstState
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		

(8) プロセスインスタンス属性の変更通知

仕様書項番	機能名	仕様書記述の API（監査情報）		
		本ワークフロー運用基盤の API（Java）		
	エンジンへの実イベント監査の通知	Prefix	M	----
		MessageID	O	WMAObjectID
		AttributeName	M	WMTAttributeName
		AttributeType	M	WMTAttributeType
		Attributelength	M	WMTAttributeLength
		Attributevalue	M	WMTAttributeValue
		PreviousAttributeLength	M	WMTAttributeLength
		PreviousAttributeValue	M	WMTAttributeValue
		Extension Number	M	WMTInt16
		Extension Type	M	WMTText
		SourceConverstaion ID	M	WMTText
		TargetConversation ID	M	WMTText
		型マッピングは、データ型定義に合わせる。		

	エンジンへの通知監査の通知	Prefix M ---- MessageID O WMAObjectID SourceActivityDefinition-BusinessName O WMAObjectName TargetProcessDefinition ID O WMAObjectID TargetProcessInstance ID O WMAObjectID TargetNodeID M WMTResourceID AttributeName O WMTAttributeName AttributeType O WMTAttributeType Attributelength O WMTAttributeLength Attributevalue O WMTAttributeValue Extension Number M WMTInt16 Extension Type M WMTText SourceConverstaion ID M WMTText TargetConversation ID M WMTText
		型マッピングは、データ型定義に合わせる。
	要求先エンジンの監査	Prefix M ---- MessageID O WMAObjectID SourceActivityDefinition-BusinessName O WMAObjectName SourceProcessInstanceID M WMAObjectID SourceActivityInstanceID M WMAObjectID SourceNodeID M WMTResourceID AttributeName O WMTAttributeName AttributeType O WMTAttributeType Attributelength O WMTAttributeLength Attributevalue O WMTAttributeValue Extension Number M WMTInt16 Extension Type M WMTText SourceConverstaion ID M WMTText TargetConversation ID M WMTText
		型マッピングは、データ型定義に合わせる。

(9) セッション管理監査情報

仕様書項番	機能名	仕様書記述の API（監査情報）
		本ワークフロー運用基盤の API（Java）
	セッション管理監査	Prefix M ---- MessageID O WMAObjectID CorrespondentDomainID M WMAResourceID CorrespondentNodeID M WMAResourceID Suffix M ----
		型マッピングは、データ型定義に合わせる。

(10) プロセス定義監査情報

仕様書項番	機能名	仕様書記述の API（監査情報）
		本ワークフロー運用基盤の API（Java）
	プロセス定義状態の変更監査	Prefix M ---- ProcessDefinitionID M WMAObjectID NewProcessDefinitionState M WMAObjectState PreviousProcessDefinitionState M WMAObjectState Suffix O ----
		型マッピングは、データ型定義に合わせる。